PROTOTIPO EXPERIMENTAL PARA LA ACTUALIZACIÓN DE UNA BASE DE DATOS DESDE UN DISPOSITIVO MÓVIL GSM ENFOCADO AL CAMPO DE LA DISTRIBUCIÓN DE HIDROCARBUROS LÍQUIDOS.

TRABAJO DE GRADUACIÓN
PREPARADO PARA LA FACULTAD
DE INGENIERÍA

PARA OPTAR AL GRADO DE:

INGENIERO ELECTRÓNICO

PRESENTADO POR:

DAVID ADRIAN ZEPEDA CORNEJO
ALDO ALBERTO PINEDA MARTÍNEZ
FRANCISCO DE LOS ÁNGELES HERNÁNDEZ AYALA

MARZO – 2006
SOYAPANGO – EL SALVADOR – CENTROAMERICA

UNIVERSIDAD DON BOSCO

RECTOR

ING. FEDERICO MIGUEL HUGUET RIVERA

SECRETARIO GENERAL

PBRO. PEDRO JOSÉ GARCÍA CASTRO, SDB

DECANO DE LA FACULTAD DE INGENIERÍA

ING. GODOFREDO GIRÓN

ASESOR DEL TRABAJO DE GRADUACIÓN

ING. JUAN CARLOS CASTRO

TUTOR DEL TRABAJO DE GRADUACIÓN

ING. EDUARDO RIVERA

JURADO EVALUADOR

ING. JUAN CARLOS CRUZ DADA

ING. WENCESLAO RIVAS

ING. ÁNGEL SORIANO

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA

JURADO EVALUADOR DEL TRABAJO DE GRADUACIÓN

PROTOTIPO EXPERIMENTAL PARA LA ACTUALIZACIÓN DE UNA BASE DE DATOS DESDE UN DISPOSITIVO MÓVIL GSM ENFOCADO AL CAMPO DE LA DISTRIBUCIÓN DE HIDROCARBUROS LÍQUIDOS.

ING. JUAN CARLOS CRUZ DADA JURADO	ING. WENCESLAO RIVAS JURADO

ING. ANGEL SORIANO
JURADO

ING. JUAN CARLOS CASTRO ASESOR	ING. EDUARDO RIVERA TUTOR

DEDICATORIA

A **Dios Todopoderoso**, por el don de la vida, y por haberme permitido alcanzar esta meta junto a una familia especial, a la que has cuidado íntegramente en el hueco de Tú mano. Gracias por los caminos angostos que de vez en cuando me has hecho transitar, ya que en ellos se ha puesto de manifiesto Tú misericordia. A ti sea toda la gloria y honra Señor.

A mis padres amados, **David Adrian Zepeda Alfaro** y **Ana Lilian Cornejo de Zepeda**, por renunciar a ellos y dedicarse a mí con tanto amor; por acompañarme y guiarme en todo momento de mi vida. Porque son el mejor ejemplo de amor, dedicación y responsabilidad que he tenido. Gracias viejitos, por la formación espiritual y humana que he recibido directamente de ustedes, así como la formación profesional que en todo momento apoyaron; estoy seguro, que es el mejor tesoro para el resto de mi vida. Por esto y por tantas otras virtudes, este triunfo les pertenece.

A mis hermanos, **Carlos Antonio Zepeda Cornejo** y **Marcela Alejandra Zepeda Cornejo**, por ser mis amigos incondicionales, acompañándome en los momentos tanto de alegría como de dificultad. También ustedes son parte de este triunfo y espero humildemente, que éste sirva de ejemplo para que ustedes puedan de la misma manera alcanzar sus propias metas.

También dedico este logro alcanzado a mis abuelitos, **Transito Rivera de Cornejo**, **Aminta Alfaro** y **Víctor Manuel Cornejo**; por sus oraciones, consejos y amor entregado. Gracias abuelitos.

AGRADECIMIENTOS

A mis compañeros y amigos de cometido, Aldo y Francisco, por su amistad y todo el empeño puesto en este trabajo. Y por compartir sus conocimientos y este triunfo conmigo.

Al Ing. Juan Carlos Castro, por el apoyo, asesoría y buena voluntad demostrada desde el inicio de este proyecto.

A Griselda Rubio, por ser esa amiga tan especial a la que admiro y respeto tanto, sin duda eres un ejemplo. Gracias por tus constantes muestras de apoyo y esa amistad tan hermosa que me has regalado.

A René Giovanni Navarro, por su amistad y ayuda desinteresada en todo momento. Gracias mi hermano.

Al P. David Panezo, sdb, Oscar Acosta y Salvador Miranda por darme la oportunidad de completar los estudios universitarios durante mi desempeño como instructor en el Colegio Salesiano Santa Cecilia.

A los ex alumnos y amigos de la promoción 2005 de Electrónica del Chaleco: Carlos Alberto Pérez, Alexis Martínez, Pedro Martínez; así como a los de la promoción 2006: Roberto José Vargas, Mario Marroquín, Miguel Ángel Melara y José Argueta. Gracias por aportar sus ideas y compartir sus cajas de elementos y herramientas.

David Adrian Zepeda Cornejo

DEDICATORIA

El trabajo de graduación esta dedicado a mis Padres que con mucho esfuerzo y cariño me brindaron su apoyo durante la carrera y a mi abuelita ya que es mi inspiración.

Dedicado también al difunto Jaggar Billar (Donde el rock nunca muere!) ya que ahí se fundaron ideas esenciales del proyecto, además de brindarnos el esparcimiento necesario para la finalización del proyecto.

Dedicado a todas las personas que de una forma directa o indirecta me ayudaron para el proyecto.

AGRADECIMIENTOS

Agradecimiento a mi familia por darme la educación, soporte, cariño y apoyo, a los alumnos del Colegio Santa Cecilia que nos prestaron los teléfonos móviles para hacer nuestras pruebas y así terminar el proyecto.

También un agradecimiento a la Familia Perdomo que me proporcionaron equipo para desarrollar la tesis y para la carrera y a la Familia Osorio que me dieron su apoyo incondicional.

Un muy especial agradecimiento a una persona que me apoyo durante la carrera y que estimo mucho y que llevo siempre en mi corazón a Susana.

A mis compañeros David y Francisco (Chico) que sin su esfuerzo no lo hubiéramos logrado y a nuestro asesor que sin su guía y enseñanzas no le hubiera logrado.

Aldo Alberto Pineda Martínez

DEDICATORIA

Dedico este trabajo de graduación a Dios Padre Todopoderoso que permitió que mi carrera fuese posible, a mi madre Marta, a mi padre Francisco y a todos mis familiares y amigos que con su valioso apoyo siempre me ayudaron a seguir adelante.

AGRADECIMIENTOS

Agradezco infinitamente a Dios Padre Todopoderoso, sin cuya fuerza, esperanza y sabiduría nada podría lograr en esta vida. Cuando en muchas ocasiones ya todo parecía estar perdido, no me abandonó, me devolvió las esperanzas, y permitió que continuara hasta alcanzar esta gran meta. Gracias porque, a pesar de no merecerlo, siempre me sostienes y nunca me desamparas.

Agradezco enormemente a esa gran mujer, que día a día ofreció su esfuerzo, su sudor, su sacrificio por mi superación sin arrepentirse ni dudarlo ni un sólo momento y de forma incondicional. A mi Madre, una de las mayores bendiciones que yo haya podido recibir en toda mi vida y por quien este logro es un hecho. Madre, te amo.

Agradezco a mi Padre, quien ha sido un importante apoyo en toda mi vida, y cuyos ánimos y consejos siempre me ayudan a continuar hasta en las circunstancias más difíciles. Papá, te amo.

Deseo agradecer a todos aquellos miembros de mi familia que de una u otra forma me ofrecieron su valiosa ayuda, me apoyaron y me animaron a lo largo de toda mi carrera. Muchísimas gracias. Los quiero.

Agradezco a todos los amigos de la familia, de Bachillerato, de la Universidad, que me han permitido contar con su apoyo, su atención y su amistad en todos estos años de estudio.

Agradezco a David y a Aldo, por permitirme haber participado en este proyecto y también por contar con su amistad. En verdad fue un gusto trabajar con ustedes.

Agradezco también al Ing. Orlando Ruiz por ayudarnos desinteresadamente al facilitarnos su equipo móvil, el cual fue una parte importante para la conclusión de nuestro proyecto.

Les estoy por siempre agradecido.

Francisco Hernández Ayala.

INDICE GENERAL

CONTENIDO PÁGINA

PRÓLOGO

I. GENERALIDADES.

En los últimos cinco años los sistemas de telefonía móvil han evolucionado en respuesta a requisitos cada vez más exigentes para mejorar tanto la calidad de la comunicación así como de ofrecer nuevos servicios, con el objetivo de mejorar la experiencia de cada uno de los usuarios.

Bajo esta evolución, en nuestros días se cuentan con teléfonos móviles que son pequeñas computadoras con la capacidad no solo de realizar la función primaria para lo cual fueron diseñados, sino también de ofrecer prestaciones como el manejo de agendas y contactos, conexión a redes de datos, descarga y ejecución de aplicaciones, servicio de navegación WAP y hasta el manejo de archivos de tipo multimedia.

Paralelo a los cambios en las redes de telefonía celular, algunas empresas desarrolladoras de software han orientado esfuerzos hacia este mercado, para brindar tecnologías compatibles que soporten las prestaciones ofrecidas a los usuarios.

Una de las prestaciones beneficiadas de este tipo de alianzas es el desarrollo de aplicaciones que aprovechando los diversos métodos para el intercambio de información, que por lo regular poseen los teléfonos móviles, la capacidad del ingreso y visualización de datos, junto con la portabilidad del propio terminal móvil, predisponen herramientas válidas para el desarrollo de aplicaciones no solo orientadas al ocio y a pequeñas utilidades de uso personal sino que a satisfacer necesidades de mayor trascendencia.

II. DESCRIPCIÓN DEL PROYECTO.

El proyecto elegido para aplicar como trabajo final de graduación, consiste en un sistema experimental que pretende actualizar una base de datos en tiempo real desde un dispositivo móvil remoto, aprovechando las versatilidades que ofrecen los nuevos terminales móviles, importados al país, en cuanto al soporte del desarrollo de aplicaciones sobre la plataforma Java2ME™ y GSM.

El desarrollo de una aplicación de este tipo, podría llevarse a cabo en cualquier proceso que involucre la manipulación y el transporte de bienes. Entre el universo de procesos a los cuales se les puede brindar una solución, se ha elegido el transporte de combustibles líquidos, para sobre éste desarrollar la aplicación descrita y de esta forma proponer una solución a la actualización en tiempo real de una base de datos desde una ubicación remota.

III. ALCANCES.

El alcance proyectado para este trabajo consiste en realizar los estudios y análisis técnicos necesarios para diseñar y construir un prototipo para la actualización de una base de datos desde un teléfono móvil GSM aplicado a un campo del que hacer económico del país.

Desde el punto de vista general, el proyecto pretende:

- Optimizar los registros en tiempo real de eventos relacionados a una base de datos desde una terminal móvil GSM.
- Evitar la manipulación de información por seres humanos y de esta forma disminuir los riesgos de corruptibilidad.
- La implementación de un sistema con tecnología y recursos disponibles en el país.

Para llevar a cabo los puntos antes mencionados deberán cubrirse las siguientes áreas:

- Sensores de flujo, circuitos de acondicionamiento, señalización y mando.
- Microcontroladores.
- Interfaces.
- Hardware y Software de teléfonos móviles GSM.
- Desarrollo de la aplicación en Java2ME™.
- Transferencia de datos en redes GSM.
- Bases de datos.

IV. LIMITACIONES.

Las limitaciones previstas durante la ejecución de este proyecto son:

- El sistema se desarrollará sobre la infraestructura del operador que soporte el terminal a emplear en el proyecto.
- El desarrollo de la base de datos no es competencia de este trabajo, por lo que para efectos demostrativos, se desarrollará una pequeña base con prestaciones reducidas.
- Disponibilidad del servicio de red del operador, bajo condiciones climáticas y ubicaciones geográficas.
- El sistema estará diseñado para la exclusiva acción de medir el volumen de hidrocarburos líquidos y no otro tipo de fluido.

INTRODUCCIÓN GENERAL

Ante la versatilidad y la mejora en las prestaciones que ofrecen los teléfonos móviles, éstos se van abriendo camino no solo para satisfacer la necesidad de comunicación entre las personas, si no que también como dispositivos centrales de sistemas capaces de realizar tareas en donde pocos quizás se puedan adaptar.

Dado el enfoque de este proyecto, el servicio de telecomunicación brindado por un teléfono móvil GSM, permite que éste se convierta en el centro del sistema que involucra diversas áreas de hardware y software, para poder brindar solución a la aplicación que pretende resolver este proyecto de final de carrera.

El documento esta divido en tres partes generales. La primera parte, formada por los capítulos del uno al tres, está dedicada a la Investigación y documentación del contexto del proyecto, así como de aquellos fundamentos técnicos necesarios que sientan las bases para el desarrollo y entendimiento del diseño y posterior construcción del sistema prototipo.

La segunda parte describe la selección de los elementos involucrados en el proceso de diseño y el proceso mismo para el desarrollo del hardware y software del sistema; esta parte la componen los capítulos cuatro y cinco.

La ultima parte formada por un solo capítulo presenta un manual de operación destinado al usuario, que describe el protocolo a realizar antes, durante y una vez finalizada la entrega del combustible a una estación cliente.

En el capítulo uno se describe los mercados de telecomunicaciones e hidrocarburos en el país, así como un resumen de la evolución de GSM.

El capítulo dos esta ligado con aspectos teóricos y técnicos del hardware involucrado en el desarrollo del proyecto, bajo una óptica que describe la secuencia de pasos que se inicia en el sensor, y posteriormente hacia las etapas de adquisición, control y mando, interfaz de

comunicación entre el microcontrolador y el teléfono móvil, teléfono móvil, concluyendo en el envió de los datos por la red de telefonía celular.

El capítulo tres describe de forma general algunos tópicos relacionados con J2ME, haciendo una introducción entre la relación de J2SE y la edición para dispositivos móviles; concluyendo en la herramienta de desarrollo de base de datos, Microsoft Access.

El capítulo cuatro, llamado diseño del hardware, describe la selección de los elementos y/o dispositivos electrónicos relacionados con el proyecto, junto con datos técnicos pertinentes para el desarrollo del mismo, y el proceso de construcción de los circuitos funcionales. Siempre enfocado a la secuencia lógica del proceso descrito en el capítulo dos. Se realiza una separación especial en el bloque funcional de la adquisición de los datos, a manera tal, de cubrir el diseño del prototipo experimental y el diseño de un sistema que pueda ser aplicado a un campo de trabajo real.

El capitulo cinco detalla el proceso de diseño del software que administra la operación del sistema completo, involucrando el programa de bajo nivel que se ejecuta en los microcontroladores, el midLET desarrollado para el teléfono móvil, la secuencia de comandos necesarios para desencapsular la información recibida en el móvil de la estación central.

Finalmente el capitulo seis, tal como se menciono anteriormente, presenta un manual para el usuario encargado de realizar la entrega del producto en una estación de servicio.

Se espera que la información técnica y científica presentada, sirva como guía para el desarrollo de posteriores aplicaciones sobre teléfonos móviles. Además de permitir la compresión del diseño, implementación y funcionamiento del prototipo presentado.

CAPÍTULO I: *ANTECEDENTES*

1.1. MERCADO DE LAS TELECOMUNICACIONES.

El mercado de las telecomunicaciones de nuestro país, ha sufrido un cambio desde los últimos años de la década de los noventas hasta el primer lustro del nuevo siglo; originado en primer lugar por los cambios de política gubernamental con respecto al sector y que seguramente continuará su evolución de crecimiento a la alza, en respuesta a la constante renovación tecnológica que generan más y más prestaciones de servicios a los usuarios, siendo ofertados por las operadoras presentes en el mercado.

1.1.1. SITUACIÓN DEL MERCADO ANTES DE LA PRIVATIZACIÓN.

Hasta el año de 1997, el mercado de telecomunicaciones en El Salvador, había sido un monopolio estatal controlado por ANTEL (Administración Nacional de Telecomunicaciones). Este organismo suministraba servicios de voz y datos relativamente modestos en prestaciones y coberturas, además de una alta relación de precio por calidad de servicio, lo cual se asoció para generar un insignificante crecimiento tanto tecnológico como en el número de usuarios.

Siendo así que hasta ese año, según el Primer Informe sobre Desarrollo Humano en El Salvador, el país contaba con una población de $5_1$908,460 habitantes y una administración total de líneas fijas operativas de 362,471; de tal forma que se gozaba de un crecimiento de 3.2 líneas telefónicas por cada 100 personas en 1992 a una teledensidad de 6.1 para finales del año 1997.

Igualmente el sector de telefonía móvil, no contaba con un clima de libre competencia y era dominado monopólicamente por el operador Telemóvil, el cual al igual que en la administradora estatal, la evolución y el crecimiento tanto en servicios, usuarios como en tecnologías era bastante deficiente, llegando a contar con 20,122

líneas móviles, conformando una teledensidad de 2.27 por cada 100 personas, según datos recopilados por la SIGET (Superintendencia General de Electricidad y Telecomunicaciones) para el año 1997.

Observando las características típicas de estructuras monopólicas-estatales y su influencia en el comportamiento de algunos sectores importantes en la dinámica económica de nuestro país, junto con los cambios económicos mundiales y el desarrollo de nuevas tecnologías, el gobierno de El Salvador tomó la decisión de aumentar su competitividad en una economía globalizada dando paso a la desmonopolización de algunos sectores económicos importantes, entre ellos el de las telecomunicaciones.

Bajo ese idea la disposición fue la privatización de dicho mercado, para cuyo fin adopto el marco legal pertinente a fin de proceder ordenada, sistemática y gradualmente hacia la venta al sector privado, tanto internacional como nacional interesado, del patrimonio de la empresa estatal.

1.1.2. PRIVATIZACIÓN DEL MERCADO DE TELECOMUNICACIONES.

Con el objetivo de que las futuras reformas y venta del mercado se hiciese en un clima de coherencia y pertinencia, el gobierno de El Salvador definió el marco jurídico sobre la reforma al sector de las telecomunicaciones en lo que se llamo: Ley de Telecomunicaciones (Decreto Legislativo No. 142 del 6 de noviembre de 1997, Diario Oficial No. 218, Tomo 337 del 21 de noviembre de 1997). El decreto 142 reformo la anterior Ley de Telecomunicaciones[1] por considerarse que sus disposiciones eran insuficientes para ejercitar una adecuada regulación y vigilancia del servicio publico de telefonía y de sus tarifas por parte del Estado.

[1] Decreto Legislativo No. 807 del 12 de septiembre de 1996, Publicado en el Diario Oficial No. 189, Tomo 333, del 9 de octubre de 1996

El otro componente importante del marco jurídico es la Ley de la Superintendencia General de Electricidad y Telecomunicaciones (SIGET), la que conforme a la ley será la entidad responsable de aplicar y velar por el cumplimiento de las normas y regulaciones establecidas en la ley y su reglamento. Estos dos instrumentos son cruciales para establecer un aceptable grado de seguridad jurídica y confianza económica y fueron la base para el proceso de privatización de ANTEL.

Una vez fue establecido el marco jurídico e institucional, para fines del proceso de privatización de la empresa estatal en el sector de las telecomunicaciones, varias empresas telefónicas internacionales se interesaron en adquirir acciones en la empresa estatal. Siendo estas, las empresas Teléfonos de México (TELMEX), Cable & Wireless de Gran Bretaña, France Telecom, Telia de Suecia, GTE y Bellsouth de los Estados Unidos; las cuales solicitaron información sobre el tamaño del mercado, marco regulatorio y proceso de venta, y realizaron visitas al país para conocer las condiciones del mercado.

El 17 de julio de 1998 se realizó, de manera exitosa a juicio de la empresa estatal, la venta del 51 % de las acciones de Internacional de Telecomunicaciones (INTEL), el segmento inalámbrico, adquiridas por Telefónica de España en un valor de US$ 41 millones. El precio base fue de de US$11.9 millones lo cual implica que el precio de la subasta casi cuadruplicó el precio inicial.

El segmento de telefonía fija de ANTEL, fue transformada a sociedad anónima denominándose CTE/ANTEL, y cuyo valor patrimonial fue establecido en US$ 527.0 millones, del cual se vendería el 51 % de las acciones. Como resultado de la evaluación de las ofertas entregadas el 27 de julio de 1998, fue declarada como ganadora la firma francesa a la que le fue vendido las acciones de CTE/ANTEL por un valor de US$ 275.1 millones.

A inicios de noviembre de 1998, France Telecom comenzó su reestructuración tanto desde el punto de vista nacional como internacional. Así también, la empresa se propuso elevar el nivel de capacidad técnica del personal con la intención de mejorar

la productividad y prestar servicios más eficientes para competir internacionalmente. Por su lado, Telefónica de España, en sociedad con inversores centroamericanos representados en el grupo Mesotel inicio operaciones en Enero de 1999, luego de haber creado la empresa Telefónica de Centroamérica en la que participa con el 51 % del capital accionario.

Con la privatización de ANTEL, de acuerdo a datos presentados por la Bolsa de Valores de El Salvador, la distribución accionaria quedó establecida así: el 51% de las acciones se vendieron al socio estratégico, el 6.1% a los trabajadores activos y pensionados y el gobierno se quedó con el 42.9% del paquete.

Para concluir, los fondos recaudados de la venta de la empresa estatal, se ha utilizado para la creación del Consorcio Educativo FANTEL, en concordancia con la Ley del Fondo Especial de los Recursos Provenientes de la Privatización de ANTEL. Dicho fondo tiene por objeto financiar la ejecución de programas y proyectos de inversión en materia de desarrollo económico y social en diversas áreas.

1.1.3. EVOLUCIÓN DEL MERCADO POST REFORMAS.

Desde el inicio de labores de las primeras operadoras que se adjudicaron la compra, de los dos segmentos de la antigua empresa de telecomunicaciones, se ha observado un crecimiento de usuarios y claras mejorías de coberturas y calidad de servicio desde el punto de vista del cliente, gracias al mercado altamente competitivo y una intervención eficiente por parte del gobierno a través de la SIGET.

De acuerdo al boletín estadístico del 2004, publicado por la SIGET, en la actualidad se encuentran en ejercicio diez operadores de telefonía fija, cuatro operadores de telefonía móvil, once operadores de larga distancia internacional y alrededor de once proveedores de servicio de Internet, lo que permite que los usuarios tengan mayores opciones de elección y una mayor cobertura en todo el país.

Como resultado de esta mayor participación de mercado se ha producido una reducción en las tarifas telefónicas beneficiando especialmente a los usuarios del servicio de llamadas internacionales hacia los Estados Unidos de Norteamérica. Sin embargo, los beneficios no se limitan a la reducción en tarifas, sino que también la teledensidad se ha incrementado sustancialmente, como producto del crecimiento acelerado de la telefonía móvil y en una menor medida la telefonía fija. (Ver gráficos 1.1.1 y 1.1.2).

El segmento de telefonía móvil y en general todos los teleservicios, ha experimentado un rápido crecimiento y aceptación en los usuarios debido a la constante renovación de prestaciones y servicio tecnológicos ofrecidos por las compañías operadoras, provocando que la SIGET renueve el Plan de Numeración Nacional y agregue un nuevo digito tanto para telefonía móvil como fija.

A excepción de Telemóvil que inicialmente contaba con una red analógica, tras los primeros meses de la apertura del mercado, inicio de forma paulatina su paso hacia una red digital de segunda generación utilizando tecnología TDMA[2]. Desde el inicio de operaciones de Telefónica, ofreció servicios de telefonía móvil basados en el uso de tecnología CDMA[3] y ofreciendo nuevos servicios auxiliares que la tecnología misma conlleva. Con el posterior ingreso de Telecom al mercado, con una red digital pero recurriendo a GSM[4], estándar experimentado en Europa con prestaciones elevadas, aumentó la oferta para el usuario. Por último, la aparición de Digicel como operador también optando por la tecnología GSM, agregaba mayor competitividad al mercado y preparando el camino hacia un nuevo cambio tecnológico.

Al término del año 2004, tanto Telefónica como Telemóvil, decidieron emigrar hacia redes bajo el estándar GSM. Logrando con esto también, que todos los usuarios puedan gozar de las mismas prestaciones que proporciona la tecnología y eligiendo al operador que más se ajuste a sus necesidades de cobertura y precio.

[2] Time Division Multiple Access.
[3] Code Division Multiple Access.
[4] Global System for Mobile Communication.

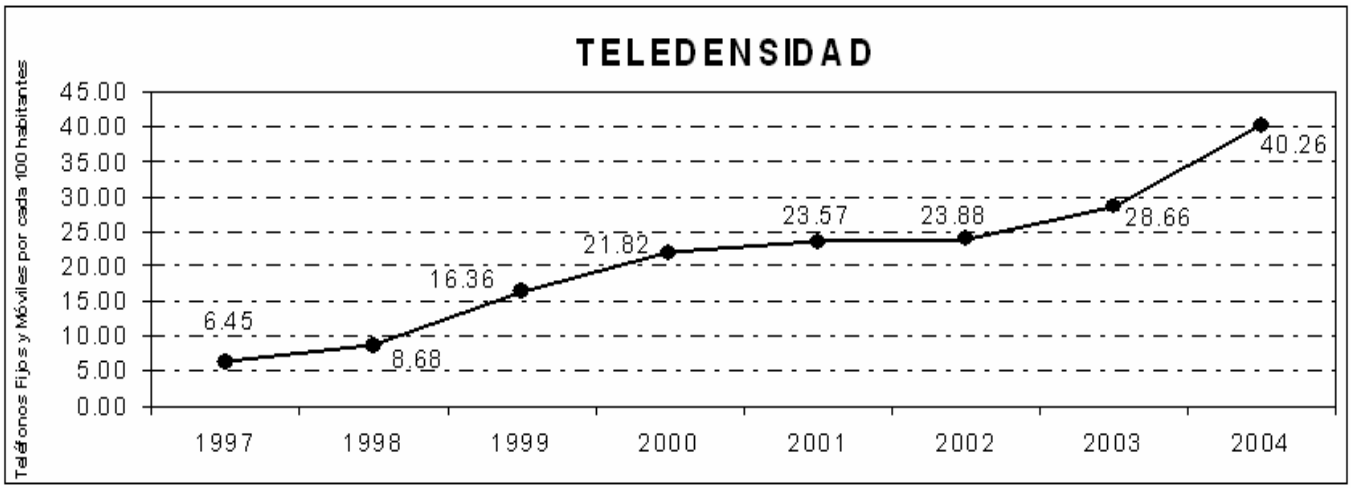


Gráfico 1.1.1 Evolución de la Teledensidad en El Salvador. [2]

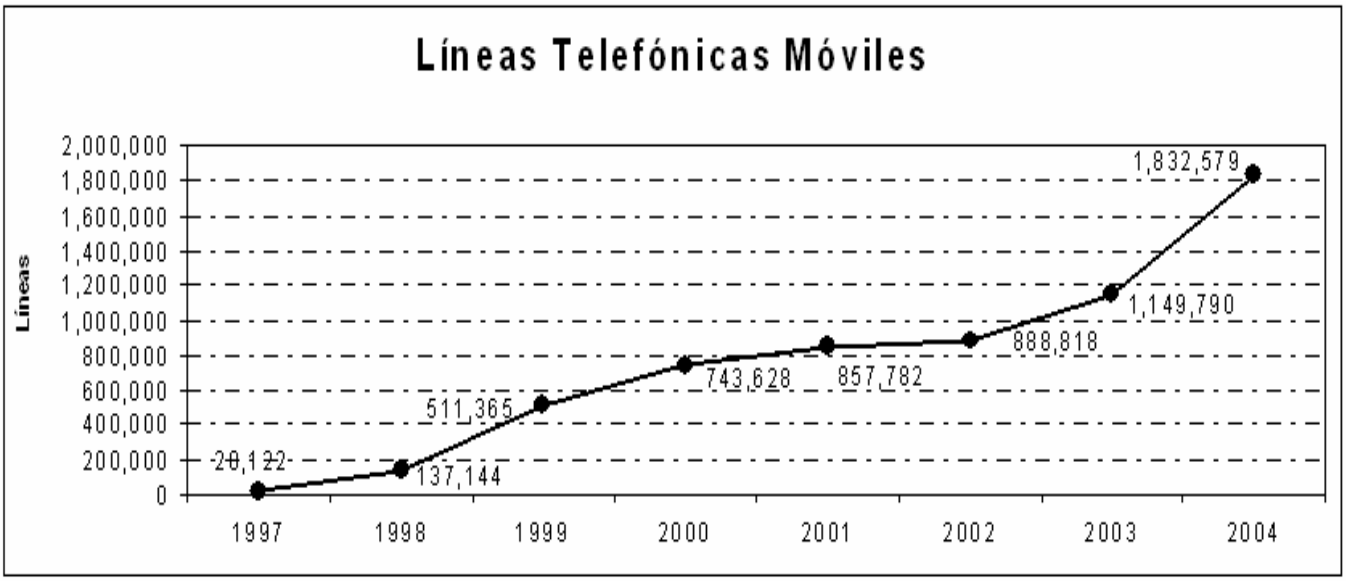


Gráfico 1.1.2 Crecimiento de Líneas Telefónicas Móviles. [2]

A inicios del año 2005, Telecom y Telemóvil, iniciaron el ofrecimiento de nuevos servicios provenientes de la actualización de sus redes GSM, hacia redes GPRS[5].

Estas nuevas prestaciones de las redes 2.5G[6], permiten a los usuarios el acceso a servidores de correo electrónico, envío y recepción de mensajería multimedia o MMS[7]

[5] General Packet Radio Service.
[6] Generación 2.5 de Telefonía Móvil.

y la descarga desde servidores de Internet, mediante navegadores WAP[8] a velocidades máximas de 115 kpbs, de contenido multimedia, juegos y aplicaciones basadas en el lenguaje de programación Java2ME.

Con estos nuevos servicios ofrecidos a los clientes, las empresas operadoras pretenden afrontar tanto tecnológica como económicamente el avance completo hacia 3G[9], y con ello mantener el clima de alta competitividad e innovación para lo cual se abrió el mercado hace unos años atrás.

1.2. EVOLUCIÓN DE GSM.

El Estándar GSM es una de las tecnologías predominantes en el mercado actual de telecomunicaciones, siendo una de sus características principales, el ser un protocolo abierto y por lo tanto susceptible a recibir aportes de empresas dedicadas al ramo, constituyéndola en una de las tecnologías pujantes en la aparición de la tercera generación en los sistemas de comunicación móvil, con un nuevo estándar, llamado 3GSM.

1.2.1. UN VISTAZO AL ESTÁNDAR GSM.

GSM aparece al inicio de la década de los noventas, tras largos años de estudio técnico, como uno de las tecnologías que dieron vida a la segunda generación de telefonía móvil junto con el estándar IS-136[10] y el IS-95A[11].

Se puede enmarcar el servicio GSM como aquel servicio portador constituido por todos los medios de transmisión y conmutación necesarios que permiten enlazar a voluntad dos equipos terminales móviles mediante un canal digital que se establece

[7] Multimedia Message Service.
[8] Wireless Access Protocol.
[9] 3ra. Generación de Sistemas de comunicación móvil.
[10] Estándar norteamericano para FDMA/TDMA.
[11] Estándar norteamericano para telefonía CDMA

específicamente para la comunicación y que desaparece una vez que se ha completado la misma.

1.2.1.1. MODO DE OPERACIÓN.

El sistema GSM permite la conexión con la red conmutada y con la ISDN[12] ofreciendo al usuario telefonía, transmisión de datos (hasta velocidades de 9.6 kbps), facsímil del grupo III, conexión a sistemas de correo electrónico (X-400) y envío de mensajes cortos (alfanuméricos) que permiten tanto su envió como su recepción desde un terminal móvil, leyéndose en este ultimo caso en el visor correspondiente.

El estándar utiliza tecnologías como FDMA[13], TDMA y FDD[14] en la banda de 900 MHz para gestionar y acceder a la comunicación entre el dispositivo móvil y la estación base. Cada estación base puede manejar 124 portadoras con separaciones de 200 kHz y proporcionar ocho canales telefónicos en una misma portadora con una compresión de voz de 13 kbps.

El uso de algoritmos potentes para verificar la autenticación, confidencialidad y asegurar el cifrado de la información tanto en la interfaz aire como a lo largo de la etapa de conmutación y transporte hasta llegar al dispositivo destino, lo convierten en una de las tecnologías más seguras en comunicación móvil.

1.2.1.2. SERVICIOS SUPLEMENTARIOS.

Los servicios suplementarios son servicios que se ofrecen para mejorar las prestaciones del servicio básico, los cuales son fundamentalmente similares a los ofrecidos en ISDN, tales como:

[12] Integrated Services Digital Network.
[13] Frequency Division Multiple Access.
[14] Frequency Division Duplex.

- Redirección de llamada entrante.
- Restricción de llamadas entrantes / salientes.
- Llamada en espera.
- Transferencia de llamada.
- Grupo cerrado de usuarios.
- Servicios multiusuarios.
- Servicios relacionados de Tarificación.
- Identificación de llamada entrante.
- Mensajería corta en modo texto o SMS[15].

1.2.1.3. TARIFICACIÓN.

A raíz que la comunicación se establece como una conexión de circuitos, entre móviles GSM, el costo del servicio se cobra en función del tiempo que dure la llamada o la transferencia de datos, existiendo la posibilidad de que los operadores ofrezcan horarios preferenciales y permitan la asignación de números frecuentes por parte de los usuarios en donde las tarifas son reducidas.

1.2.2. PREPARACIÓN DE GSM HACIA 3GSM.

Tanto HSCSD[16], GPRS y EDGE[17], se presentan como el conjunto de tecnologías de 2.5G que preparan técnicamente el camino desde GSM hacia el avance de GSM de tercera generación.

La elección que un gran número de operadoras de telefonía móvil occidentales, con redes basadas en GSM han decido tomar para la evolución hacia 3GSM, consiste en dar una paso hacia GPRS de tal forma que les permita estar preparados técnicamente

[15] Short Message Service.
[16] High Speed Circuit Switched Data
[17] Enhanced Data Rate for GSM Evolution

hacia los nuevos servicios que arrastra la tecnología de 3G y afrontar desde el punto de vista económico el cambio total de la infraestructura.

1.2.2.1. FILOSOFÍA DE GPRS.

GPRS, es un sistema de telefonía móvil que está basado en la conmutación de paquetes sobre la red GSM que se usa actualmente. GPRS es una tecnología estandarizada por el ETSI (European Telecommunications Standard Institute) como parte de GSM Fase 2+ que permite la transmisión de datos a velocidades de hasta 115kbit/s, unas 10 veces más rápidas que en GSM y que ofrece soporte para el acceso a Internet y correo electrónico.

La nueva tecnología GPRS se basa en la optimización de la tecnología GSM utilizada hasta ahora para las comunicaciones móviles, a la que se añaden capacidades adicionales de transmisión de datos. Esto supone la utilización de la misma infraestructura de red, con las ventajas que ello conlleva, como aprovechar la experiencia adquirida y poder disponer de un rápido despliegue de red que permite una cobertura geográfica similar a los niveles actuales alcanzados en GSM.

1.2.2.2. MODO DE OPERACIÓN.

La clave de GPRS se basa en el diferente tratamiento que la red hace de la voz y los datos, lo que permite que con la misma capacidad de red se puedan obtener, en transmisión de datos, rendimientos muy superiores a los conseguidos con GSM.

En GSM, el tráfico de voz y datos, se realiza mediante la conmutación de circuitos, técnica utilizada en telefonía fija y que ocupa mientras dure la llamada, un canal especifico para concretar la comunicación. Mientras que en GPRS, el tráfico de datos se realiza bajo la conmutación de paquetes, similar a la utilizada en redes cableadas de datos para computadoras, lo que significa que la información es

fraccionada en el dispositivo móvil origen y transmitida en pequeños paquetes, siendo reagrupada posteriormente en el equipo destino.

Con esta técnica de conmutación utilizada por GPRS, se utiliza de forma eficiente los canales disponibles en el área de cobertura donde este ubicado el teléfono móvil, debido a que son utilizados únicamente cuando es necesaria la transferencia de información.

Dadas las características presentadas anteriormente, GPRS ofrece acceso instantáneo a protocolos X.25[18] e IP[19] y de esta forma se obtienen dispositivos móviles con asignación de direcciones IP con una conexión activa permanente mientras el dispositivo esta encendido, llamado técnicamente como “always on”.

1.2.2.3. CARACTERÍSTICAS DE LA TECNOLOGÍA.

De la forma de operación de la tecnología, se pueden destacar las siguientes características:

- Compatibilidad total con todos los servicio prestados por GSM.
- Conexión permanente para la transferencia de datos.
- Aumento en la velocidad de transmisión de datos.
- Soporte para mensajería multimedia MMS.
- Soporte para la conexión de Internet.
- Separación en el modo de facturación: se cobrará por volumen de paquetes enviados/recibidos para la transferencia de datos, por la cantidad de tiempo de utilización del canal establecido para la comunicación de voz y una tarifa especial para cada tipo de mensaje enviado ya sea SMS o MMS.

[18] Protocolo de comunicación para redes cableadas de datos.
[19] Internet Protocol.

1.2.3. ¿QUÉ SE ESPERA DE 3GSM?

Según GSMA (GSM Association), 3GSM esta siendo desarrollado por un grupo de operadores bajo el nombre de 3GPP (The 3rd Generation Partnership Project). Dicha tecnología será el eslabón final desde GSM, pasando por GPRS y EDGE; para obtener un estándar abierto que utilizará W-CDMA[20] como técnica de acceso en la interfaz aire.

Como 3GSM se está construyendo alrededor del núcleo de GSM, se espera que todos aquellos servicios de voz y suplementarios prestados por el estándar de 2G y 2.5G se mantengan y adicionalmente ofrezca soporte al acceso de redes con velocidades cercanas a los 2 Mbps, beneficiando nuevas utilidades como la de telecomunicación multimedia en tiempo real, y el comercio electrónico móvil (m-Commerce).

1.3. MERCADO DE HIDROCARBUROS.

El mercado de hidrocarburos en El Salvador sufrió un cambio de una época donde, tanto la importación y la comercialización como los precios de venta estaban completamente controlados, hacia un libre mercado donde el Estado interviene en los aspectos normativos y de regulación del abastecimiento, seguridad, calidad y entrega exacta de los combustibles comercializados, protección y conservación del medio ambiente, entre otros.

Los precios de todos los productos están liberalizados en todas las diferentes etapas de comercialización, con excepción del gas licuado de petróleo para consumo doméstico en envases de 35, 25, 20 y 10 libras que tiene precio de venta fijado por medio de Acuerdo Ejecutivo en el Ramo de Economía por ser un producto subsidiado.

[20] Wideband-CDMA

En este caso se establecen los precios máximos del producto subsidiado que factura el importador y refinador local a las compañías mayoristas, por medio del Sistema de Precios de Paridad de Importación; para los otros productos se calculan precios de referencia, utilizando el Sistema ya citado y tomando como base el mercado de la Costa del Golfo de los Estados Unidos de América. Los precios de referencia no son de cumplimiento obligatorio para las compañías mayoristas.

Con dicha apertura, cualquier persona natural o jurídica puede importar y exportar, debiendo únicamente estar inscrita en el Registro de Comercializadores de Productos de Petróleo que esta bajo el control de la Dirección de Hidrocarburos y Minas, del Ministerio de Economía.

1.3.1. COMPETIDORES.

El mercado está totalmente integrado por quince empresas del sector privado que se encargan de importar y comercializar productos derivados del petróleo, tales como gasolina de 95 octanos, gasolina de 90 octanos, diesel (No. 2), kero/turbo, fuel oil (No. 6), asfaltos cutback/penetración y gas licuado de petróleo (GLP). Dentro de estas empresas, se encuentra la Refinería Petrolera Acajutla S.A. de C.V (RASA de C.V.) que importa productos terminados y además refina productos a partir de una mezcla de petróleo crudo y reconstruido. Las empresas importadoras y comercializadoras clasificadas por tipo de combustible que actualmente operan en el país, se presentan en la tabla 1.3.1.

En lo que respecta a los hidrocarburos líquidos, específicamente gasolinas y diesel, de acuerdo a datos de la Dirección de Hidrocarburos y Minas, hasta el 30 de mayo del 2005 existen un total de 385 estaciones de servicio que en conjunto, durante el ejercicio del año recién pasado comercializaron $8_1$219,274 barriles de combustible.

EMPRESA	TIPO DE COMBUSTIBLE					
	Gasolina 95 octanos	**Gasolina 90 octanos**	**Diesel**	**Kerosina**	**Fuel Oil**	**GLP**
RASA de C.V.	▪	▪	▪	▪	▪	▪
Esso Standard Oil S.A., Ltd	▪	▪	▪	▪	▪	▪
Texaco Caribbean Inc.	▪	▪	▪	▪	▪	
Shell El Salvador S.A.		▪	▪	▪	▪	
Puma El Salvador S.A.	▪	▪	▪			
El Paso Technology El Salvador S.A. de C.V.					▪	
Tropigas de El Salvador S.A.						▪
Terminales de Gas del Pacífico S.A. de C.V.						▪
Coinver S.A. de C.V.						▪
ELF Gas El Salvador S.A. de CV.						▪
ZETA Gas de El Salvador S.A. de C.V.						▪
Distribuidora Salvadoreña de Lubricantes y Combustibles, S.A. de C.V.	▪	▪	▪	▪	▪	
Distribuidora Salvadoreña del Petróleo, S. A. de C.V.					▪	
DIPROPESA, S.A. de C.V.	▪	▪	▪			
SEVGASA, S.A.	▪	▪	▪			

Tabla 1.3.1 Resumen de empresas y productos. [9]

De acuerdo con la información extraída de las tablas 1.3.2 y 1.3.3 el subsector de hidrocarburos líquidos, presenta un mercado que se mantiene constante en la demanda del producto a pesar de las continuas alzas en los precios de los mismos.

Las tablas 1.3.2 y 1.3.3 muestran la desagregación de las estaciones de servicio por compañías y departamento, y el consumo de combustibles típicos durante el periodo 2000 – 2004, respectivamente.

Departamento	Shell	Esso	Texaco	Bandera Blanca	Total
San Salvador	37	29	34	16	116
La Libertad	11	10	4	14	39
Cuscatlán	3	1	2	0	6
La Paz	6	3	5	7	21
Cabañas	2	2	0	1	5
San Vicente	1	3	2	8	14
Santa Ana	13	7	10	10	40
Ahuachapán	4	2	3	6	15
Chalatenango	4	1	4	2	11
Sonsonate	11	5	6	9	31
Usulután	7	2	9	6	24
San Miguel	12	8	8	6	34
Morazán	3	1	1	4	9
La Unión	5	4	4	7	20
TOTAL	**119**	**78**	**92**	**96**	**385** [21]

Tabla 1.3.2 Estaciones de servicio de combustible. [9]

CONSUMO GASOLINAS Y DIESEL (BARRILES)[22]					
PRODUCTOS	**2000**	**2001**	**2002**	**2003**	**2004**
GASOLINA 95 OCTANOS	1,144,953	847,947	1,122,255	1,296,971	1,422,280
GASOLINA 90 OCTANOS	1,829,007	2,227,396	2,168,528	2,141,603	2,088,703
DIESEL	4,875,184	4,704,247	4,698,759	4,864,455	4,708,290
TOTAL	**7,849,144**	**7,779,591**	**7,989,538**	**8,285,029**	**8,219,274**

Tabla 1.3.3 Demanda de combustible periodo 2000 – 2004. [9]

[21] Incluye 16 estaciones cerradas temporal o definitivamente.
[22] Incluye combustible utilizado para generación de energía eléctrica.

1.3.2. REGULACIÓN.

El marco regulatorio y normativo lo aplica la Dirección de Hidrocarburos y Minas del Ministerio de Economía, entre las funciones asignadas a la citada Dirección y que están relacionadas con los hidrocarburos se encuentran las siguientes:

- Elaborar y actualizar las normas e instrumentos legales, técnicos y administrativos del sector minero y del subsector hidrocarburos.
- Establecer los mecanismos de coordinación para el adecuado abastecimiento del petróleo y derivados.
- Supervisar y controlar el abastecimiento y comercialización de los hidrocarburos.
- Controlar la calidad y cantidad de los combustibles, así como regular los aspectos de seguridad industrial en la importación, exportación, comercialización, manejo y almacenamiento de hidrocarburos en el país.
- Elaboración e implementación de normas y especificaciones técnicas de los sectores energéticos y minero.
- Mantener estadísticas actualizadas sobre el petróleo y derivados.
- Fomentar la inversión privada, extranjera y nacional y promover el uso de combustible mejorados para la protección del medio ambiente.
- Conocer y resolver sobre los diferentes casos de carácter jurídico y técnico que de acuerdo a la legislación vigente se presenten.
- Representar al país en todos aquellos eventos a nivel nacional e internacional relacionados con el sector minero y subsector hidrocarburos.

El marco legal del sector hidrocarburos comprende:

- Disposiciones del Reglamento Interno del Órgano Ejecutivo en el Ramo de Economía.
- Ley Reguladora del Depósito, Transporte y Distribución de Productos de Petróleo y su Reglamento de Aplicación.
- Ley de Protección al Consumidor y su Reglamento.
- Ley del Consejo Nacional de Ciencia y Tecnología, CONACYT.

- Normas Salvadoreñas Obligatorias, (NSO); así como en los Acuerdos, Resoluciones e Instructivos emitidos con base en las citadas Leyes.
- Aplicación de la Ley del Medio Ambiente y su Reglamento.
- Ley del Impuesto a la Transferencia de Bienes Muebles.
- Ley de Prestación de Servicios y su Reglamento.

CAPÍTULO II: *HARDWARE*

2.1. SENSORES, ACONDICIONADORES DE SEÑAL Y ACTUADORES.

En los sistemas de medición, supervisión, y control de procesos, como el presentado en la figura 2.1.1, es común identificar una mezcla de cuatro bloques funcionales que se encargan de adquirir, adecuar, procesar y actuar como respuesta a estímulos externos de algún tipo de variable física.

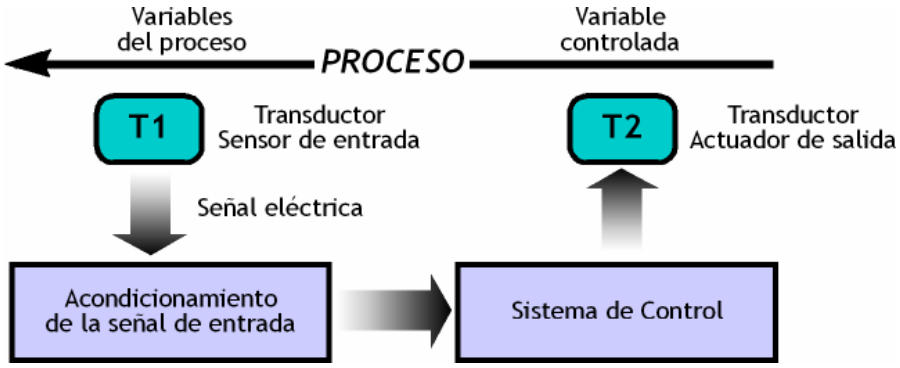


Figura 2.1.1 Esquema general para el control de procesos. [10]

El primero y último bloque son llamados transductores, en general un transductor es un dispositivo capaz de realizar la conversión de un tipo de energía física a otra. El primer bloque entonces se nombra como transductor de entrada o sensor, el cual es un tipo de transductor que acepta estímulos del proceso que se está supervisando y únicamente responde con cambios de alguna magnitud eléctrica. El último bloque del sistema es conocido como transductor de salida o actuador, y son dispositivos diseñados para aceptar una señal eléctrica del sistema de supervisión y convertirla en algún tipo de energía física adecuada para interactuar con el mundo externo.

El bloque de acondicionamiento se encarga de entregar las señales eléctricas provenientes del sensor, en una forma que, a petición del bloque de control, cumpla con ciertas características para su posterior procesamiento.

El control, independiente de su naturaleza analógica o digital, aceptará las señales eléctricas provenientes del circuito acondicionador, en las cuales se basará para tomar las decisiones y emitir señales de salida con el objetivo de interactuar con el mundo físico.

2.1.1. SENSORES ELECTRÓNICOS.

Los sensores electrónicos son un subconjunto de los dispositivos de transducción, que proporcionan una salida útil en respuesta a un mensurando especifico. El mensurando es una cantidad, propiedad o condición física que se mide y la salida es la cantidad eléctrica, producida por un transductor, que es función del mensurando especifico. Aunque en la práctica la respuesta eléctrica de la señal de salida, se ve afectada por otras dos fuentes que a la larga perturban la respuesta del estimulo propiamente dicho.

En la figura 2.1.2 se muestra un esquema que denota la modificación de la señal de salida, ante señales perturbadoras como el ruido (*Sn*) y las señales modificadoras (*Sm*) al sumarse con la respuesta producida por la interacción misma de la magnitud física con el sensor (*Sd*).

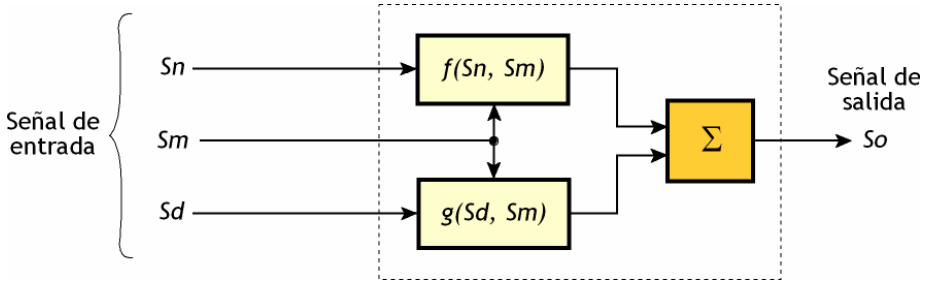


Figura 2.1.2 Señales condicionantes en la operación de un sensor. [10]

Las funciones *f(Sn, Sm)* y *g(Sd, Sm)* describen el proceso de conversión, que transforma la señal de entrada a una forma apropiada para la salida. Simplemente, las funciones *f* y *g* indican que la señal de salida es alguna función del estimulo de entrada y, de alguna manera, es proporcional a éste.

El ruido de entrada representa cualquier señal ajena al proceso a la cual responde el sensor. El estimulo modificador, representa las señales externas que pueden causar alteraciones en la respuesta del transductor. Estas señales alteran tanto la señal deseada como la señal no deseada.

2.1.1.1. CARACTERÍSTICAS.

Algunas características intrínsecas y extrínsecas que intervienen en la calidad de la medición hecha con un sensor, y en general sobre su rendimiento a lo largo de su vida útil, son la sensibilidad, precisión, linealidad, repetibilidad, confiabilidad y costo. Estas características obviamente también influyen en el diseño del circuito de instrumentación o de adquisición asociado con el sensor.

2.1.1.1.1. FUNCIÓN DE TRANSFERENCIA.

Todo tipo de transductor tiene una relación ideal salida mensurando, descrita por una ecuación teórica o por una representación tabular (numérica) o gráfica.

Esta característica ideal de transferencia es lineal en muchos casos; es decir la pendiente de la recta descrita por la función es llamada función de transferencia del sensor. En el caso de una característica ideal no lineal, la relación de transferencia se emplea algunas veces para describir el comportamiento del dispositivo.

2.1.1.1.2. SENSIBILIDAD.

La sensibilidad de un transductor se define como la proporción de cambio en la señal de salida, conforme varía la señal deseada a la entrada. En forma matemática,

$$\frac{dS_o}{dS_d} = K + \frac{dS_n}{dS_d} + \frac{dS_m}{dS_d} \qquad \text{Ecuación 2.1}$$

La ecuación indica que la sensibilidad es igual a la sumatoria de la función de transferencia del sensor, y las razones de cambio de las señales de ruido y modificadora, con respecto a la señal deseada. La ecuación considera además, que no solo la relación del estimulo con la salida debe ser el centro de atención, si no que también la influencia de las señales externas antes mencionadas.

2.1.1.1.3. PRECISIÓN Y LINEALIDAD.

Cualquier dispositivo de medición inevitablemente perturba el proceso que se está supervisando, de manera que nunca se obtienen lecturas reales. Sin embargo, se puede manejar las mediciones obtenidas si se conoce con cuanta precisión fueron hechas.

La precisión, por lo tanto, se relaciona con que el rango de exactitud elegido para el proceso de medición, también conocido como error de medición, y corresponde al margen máximo permitido entre el valor medido y el real. La habilidad para mantener la precisión durante todo el proceso depende de la linealidad del sistema, esto es, de cuán cercana es la respuesta directamente proporcional al estimulo.

La gráfica de la figura 2.1.3 presenta un conjunto de lecturas tomadas con algún incremento sobre diversos valores o estímulos.

Figura 2.1.3 Definición de error y linealidad de un sistema [10]

Se presenta una dispersión estadística de las lecturas, y el poder considerar lineal la respuesta dependerá de cuánta precisión se exija. Aceptar un margen o límite de error garantiza que todas las lecturas se encuentren dentro de líneas paralelas, y así poder afirmar que la respuesta es lineal, pero esto es significativo sólo si el límite de error es suficientemente pequeño.

Un método común para especificar la linealidad de un transductor, es haciendo referencia a una curva promedio de calibración, que se toma como la línea recta que mejor se ajusta a un conjunto disperso de lecturas. El criterio que suele utilizarse es el de mínimos cuadrados. En este enfoque la línea de calibración se traza de manera que la suma de los cuadrados de las diferencias verticales entre las lecturas y la línea se reduzca al mínimo.

En la práctica de la ingeniería, la precisión de un instrumento se define en términos de la mayor desviación horizontal a partir de la línea de calibración, y suele expresarse como un valor porcentual de la lectura de escala total.

2.1.1.1.4. CONFIABILIDAD.

Además de las características que antes se han mencionado, la confiabilidad es otro factor importante, ya que se espera que el transductor genere respuestas virtualmente idénticas al mismo estimulo de entrada durante toda su vida de

servicio. De hecho, se puede decir que la vida de servicio de un transductor se define como el periodo en que continué funcionando con precisión (dentro de límites predeterminados). A menudo la confiabilidad se relaciona con el costo del dispositivo, y la especificación de su vida de trabajo se convierte en un ejercicio de costo eficiencia.

El fabricante proporciona la evaluación de confiabilidad en términos estadísticos, ya que en general es impractico medir todos los parámetros de cada transductor. La confiabilidad es, entonces, la probabilidad de que el transductor funcione en forma satisfactoria dentro de sus especificaciones.

A la larga, todos los dispositivos fallan y el patrón de fallas en la vida útil de un lote de dispositivos idénticos puede resumirse en un diagrama llamado, por su forma, diagrama de tina de baño, como el mostrado en la figura 2.1.4.

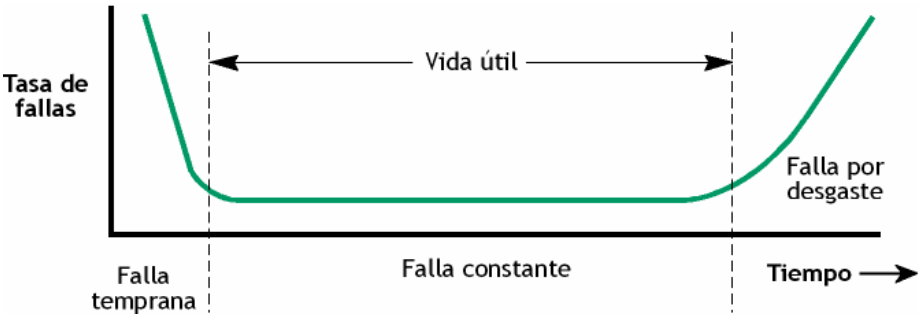


Figura 2.1.4 Evolución de las fallas a lo largo del tiempo de vida. [10]

La falla temprana suele llamarse de calentamiento y, cuando la confiabilidad es muy importante, puede ser impuesto operando los dispositivos en condiciones apropiadas por un periodo suficientemente extenso para eliminar los componentes débiles, antes de que se utilicen los restantes en un sistema. La vida útil se extiende en todo el periodo de falla constante, y el propósito es reducir los defectos al nivel mas bajo posible. El periodo durante el cual los dispositivos operan satisfactoriamente varía mucho de uno a otro, y todo lo que se puede hacer es proporcionar algunas indicaciones del tiempo promedio en que la operación será satisfactoria.

2.1.1.1.5. OTROS PARÁMETROS.

- Campo de medición: Corresponde al margen entre los valores mínimo y máximo de la magnitud física medida por el transductor.

- El valor de FSO (Full Scale Output): Atañe al valor de fondo de escala, y es la diferencia entre las tensiones de salida del transductor, correspondientes a los valores límites del campo de medición.

- Error de linealidad: Es el desplazamiento de la constante de proporcionalidad entre el valor de entrada y el de salida. Éste se expresa en porcentaje del valor máximo de salida.

- Velocidad de respuesta: Incumbe a la rapidez con la que la magnitud de salida responde a las variaciones de la magnitud de entrada.

2.1.1.2. CLASIFICACIÓN DE LOS SENSORES.

Existen diversas formas de clasificar los dispositivos electrónicos, unas que se consideran bastante pertinentes para ser aplicadas a los sensores, son las basadas en los siguientes tres aspectos: la respuesta de salida del transductor ante un estimulo, la naturaleza de la señal de salida, y el tipo de variable física a medir.

2.1.1.2.1. CLASIFICACIÓN POR LA RESPUESTA DE SALIDA.

Ésta clasificación divide en dos a los sensores:

- Sensores Pasivos: En estos sensores la acción del mensurando produce un cambio en la resistencia, inductancia o capacitancia, según el principio en el

que se basa el elemento. Estos dispositivos requieren de una fuente eléctrica externa para la excitación.

- Sensores Activos: Los sensores activos, son los que entregan una diferencia de potencial producto de algunos fenómenos tales como el termoeléctrico o piezoeléctrico. La salida autogenerada suele ser a bajo nivel y requiere amplificación. También los sensores basados en materiales semiconductores se agrupan baja esta clasificación.

2.1.1.2.2. CLASIFICACIÓN POR LA NATURALEZA DE LA SEÑAL DE SALIDA.

Esta clasificación, también subdivide a los sensores en dos tipos:

- Analógicos: La respuesta de estos sensores, junto con el circuito de instrumentación, se define como una salida que es una función continua del mensurando. Esta función continua puede ser palpable como una variación de voltaje o de frecuencia.

- Digitales: Por el tipo de aplicación al cual están orientados o por razones de diseño, algunos transductores proporcionan señales discretas con dos únicos posibles estados, ante la influencia de una variación física.

2.1.1.2.3. CLASIFICACIÓN POR EL TIPO DE VARIABLE FÍSICA MEDIBLE.

En la actualidad y gracias al mejoramiento de las técnicas de fabricación de los elementos electrónicos, es posible la integración del sensor junto con el circuito necesario para la manipulación de la respuesta, lo que permite poder

medir o supervisar casi cualquier variable física que rodea nuestro entorno. Algunos de los sensores, son:

- Sensores para movimientos mecánicos de sólidos.
- Sensores para cantidades de mecánica de fluidos.
- Sensores para cantidades térmicas.
- Sensores para cantidades acústicas.
- Sensores para cantidades ópticas e infrarrojas.
- Sensores para radiación nuclear.
- Sensores para campos magnéticos.
- Sensores electroquímicos.
- Sensores biométricos.
- Sensores para la adquisición de imágenes.

2.1.2. SENSORES ELECTRÓNICOS PARA LA MEDICIÓN DE FLUJO VOLUMÉTRICO.

El mensurado denominado gasto, caudal o intensidad de flujo puede en realidad ser alguna de tres cantidades físicas diferentes:

- La velocidad lineal del fluido en un punto específico, una cantidad vectorial con magnitud y dirección, medida con respecto a una referencia que puede ser estacionaria o móvil.

- El gasto volumétrico a través de un área transversal, que es la integral de superficie del gasto lineal sobre el área.

- El gasto de masa a través de un área transversal, que es la integral de superficie de la velocidad multiplicada por la densidad.

En la mayor parte de los casos el flujo esta confinado, ya sea en canales abiertos como ríos y tuberías parcialmente llenas, con una frontera de líquido libre o canales cerrados (tuberías llenas), y el caudal por medir es unidireccional, a lo largo del eje del canal.

Con algunas excepciones, por lo general siempre se está interesado en el gasto volumétrico o de masa. Aunque en muchas aplicaciones se requiere una medición del flujo de masa, no es fácil detectarlo y medirlo, y la mayor parte de los flujómetros son volumétricos. Así, el flujo másico debe determinarse mediante una medición o cálculo simultáneo de la densidad.

Virtualmente todos los flujómetros volumétricos se basan en la detección de la velocidad de flujo, y se calibran en términos del volumen. En algunos tipos de medidores la velocidad del flujo se detecta o muestrea en un punto, o en una pequeña área, y para determinar el gasto volumétrico es necesario conocer el perfil de velocidad, es decir, la distribución de la velocidad del fluido en la sección transversal del canal. Tal lectura del flujómetro debe corregirse en caso de que el perfil de velocidad sea variable o si no es idéntico a aquel en el que se basa la calibración.

En otros detectores de flujo la salida representa un promedio de velocidades sobre el perfil. Algunos flujómetros miden directamente el volumen.

El perfil de velocidad en canales cerrados se determina por medio del número de Reynolds, una relación adimensional definida como:

$$R_e = \frac{V \cdot D \cdot \rho}{\mu} \qquad \text{Ecuación 2.2}$$

Donde V = velocidad del flujo, m/s

D = diámetro de la tubería, m

ρ = densidad, kg/cm3

μ = viscosidad dinámica, Pa·s

Para números de Reynolds inferiores a 2,000, el flujo es laminar y el perfil de velocidad es parabólico, y para números de Reynolds mayores de 10,000, el flujo es turbulento y el perfil de velocidad es esencialmente uniforme en todo el diámetro. La transición de flujo laminar a turbulento es gradual, y el patrón de flujo en el intervalo de números de Reynolds es entre 2,000 a 10,000, rango que no se encuentra definido con claridad.

En la mayor parte de las instalaciones de flujómetros, es aconsejable que la localización del detector de flujo esté precedida y seguida de varios diámetros de tubo recto de sección transversal uniforme, a fin de que el perfil de velocidad para el cual se haya calibrado el flujómetro se desarrolle por completo.

Dado que tanto la densidad como la viscosidad son características de un material específico, un flujómetro calibrado para un líquido puede producir errores cuando se utilice con otro líquido. Además, los cambios de temperatura y presión también provocan cambios en el número de Reynolds, por lo que afectan la calibración del flujómetro.

La exactitud de la medición de flujo también se basa en un conocimiento a priori acerca de la composición física del fluido. Por ejemplo, en la lectura de la salida volumétrica de un flujómetro basado en un detector de velocidad de flujo se supone que toda la tubería está llena de líquido; la lectura será errónea si existen burbujas de gas en el líquido.

2.1.2.1. FLUJÓMETROS CON PARTES MÓVILES.

Varios detectores de flujo operan con partes móviles expuestas al fluido, lo cual los hace inútiles en el caso de medir sustancias corrosivas. Miden el flujo en base a un desplazamiento o velocidad angular que dependen del gasto.

2.1.2.1.1. FLUJÓMETRO DE DESPLAZAMIENTO POSITIVO.

Posee un rotor que consta de cámaras de volumen fijo. El ángulo de rotación es directamente proporcional al volumen del fluido. Estos medidores suelen emplearse para medir el volumen total, en vez del gasto, por ejemplo en las mediciones de agua y combustible. Su exactitud es elevada, particularmente en los tamaños grandes, donde se obtiene el 0.1 % de exactitud en líquidos y el 0.5 % en gas.

El flujómetro de desplazamiento positivo es en realidad un motor de fluido de desplazamiento positivo diseñado para fuga, fricción e inercia pequeñas. Un esquema relacionado es la bomba de medición, un dispositivo de desplazamiento positivo especialmente diseñado que mide el flujo mientras opera como una bomba; su exactitud es del orden del 1 %.

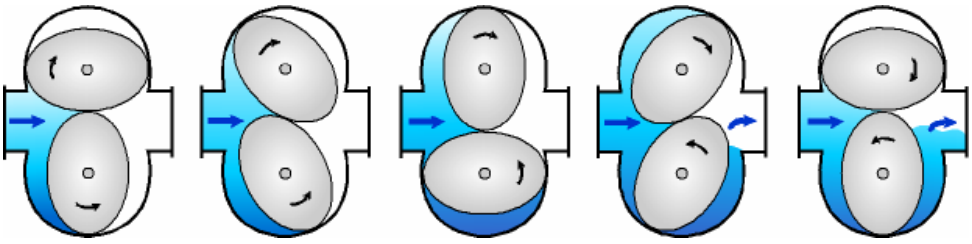


Figura 2.1.5 Flujómetro de desplazamiento positivo [13]

2.1.2.1.2. FLUJÓMETRO DE TURBINA.

Posee un rotor en el tubo, que es accionado por el fluido. La velocidad angular es proporcional al gasto. Para detectar la velocidad suele emplearse un captor magnético, de modo que la salida eléctrica es de la forma de una frecuencia (FM) o de una variación de pulsos. Este flujómetro se utiliza con fluidos limpios, y su exactitud es del orden del 0.5 % del gasto real.

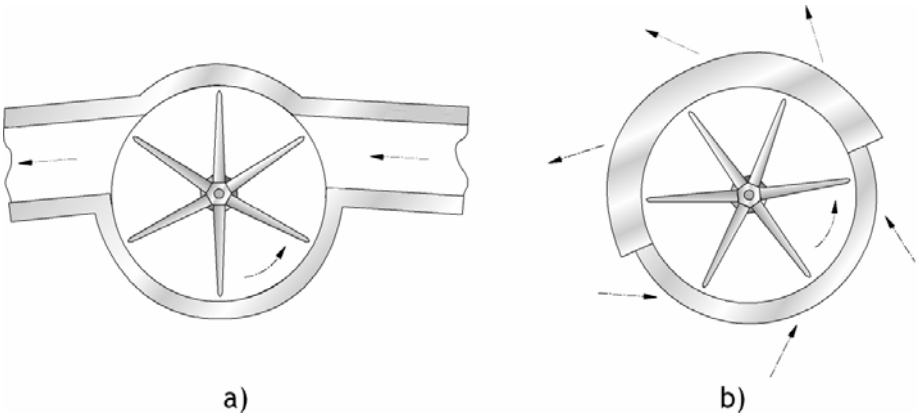


Figura 2.1.6 Flujómetro de turbina (a) de un chorro (b) varios chorros. [13]

2.1.2.2. FLUJÓMETROS MAGNÉTICOS.

Este flujómetro se basa en la ley de Faraday. Sus elementos fundamentales, que se muestran en la figura 2.1.7, son: un fluido con conductividad eléctrica, un tramo de tubo recto aislado, un conjunto de bobinas electromagnéticas a fin de producir un campo magnético perpendicular a la dirección de movimiento del fluido, y un par de electrodos localizados sobre el tubo, con sus ejes perpendiculares tanto al campo magnético como al fluido en movimiento.

Dado que la densidad de flujo depende de la corriente que fluye a través del electroimán, la salida es relaciométrica; es decir, la información de gasto debe determinarse a partir de la relación del voltaje de salida del flujómetro *E* entre un voltaje de referencia *Er*, obtenido de la corriente de la bobina electromagnética. La medición básica es la de una velocidad promedio del fluido, independiente de variaciones en la presión, densidad y viscosidad.

La calibración en términos del flujo volumétrico puede realizarse con exactitud sobre amplios intervalos de perfil de velocidad mediante la conformación adecuada del campo magnético. El voltaje de salida del detector de flujo es bastante pequeño, del orden de los milivoltios, por lo que el acondicionamiento cuidadoso de la señal

ha sido la clave para el diseño con éxito de flujómetros magnéticos. La exactitud es del orden del 0.5 % de la escala completa.

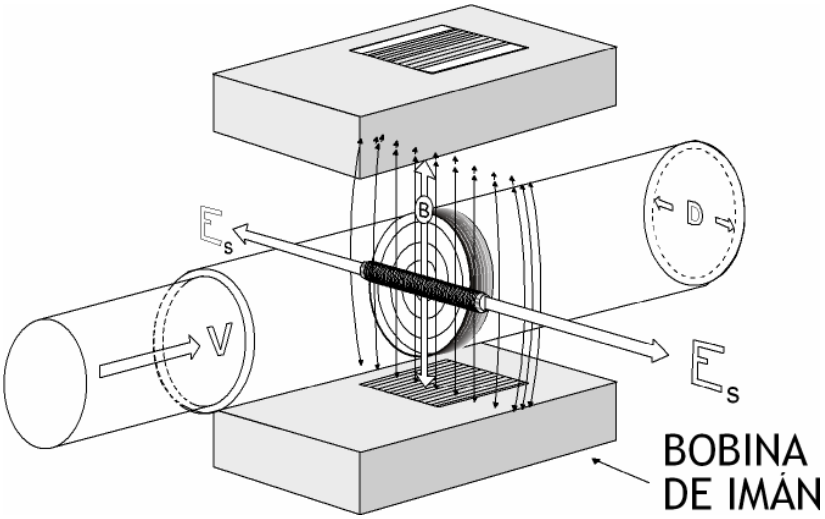


Figura 2.1.7 Flujómetro Magnético. [11]

El voltaje de salida es:

$$E = B \cdot D \cdot \nu \times 10^{-4} \quad \text{V} \qquad \text{Ecuación 2.3}$$

Donde B = densidad de flujo, T

D = diámetro del tubo, m

υ = velocidad del fluido, m/s

La conductividad del fluido afecta el rendimiento global, debido a que influye en el comportamiento del detector de flujo como la resistencia de la fuente para el voltaje de salida, Cuanto más pequeña sea la conductividad, mayor será la resistencia interna y más difícil será el acondicionamiento de la señal. Diseños actuales permiten la medición de fluidos hasta 10 μS/m (0.1 μS/cm), lo cual incluye una amplia variedad de líquidos, entre ellos, los solventes orgánicos; pero no así los gases, vapores o hidrocarburos.

La excitación del campo magnético puede ser por CD, CA o por pulsos. La excitación por CD en la mayor parte de los líquidos produce polarización, lo que da por resultado una gran salida constante de CD, sobre la cual se sobrepone la pequeña salida dependiente de la velocidad. De este modo, la excitación por CD no es práctica, excepto en la medición de flujo de metales líquidos.

La excitación del campo por CA elimina el problema de la polarización, pero da por resultado una salida significativa de flujo cero (nulo) debido a corrientes parásitas en el fluido mismo y a voltajes captados de circuitos externos.

En la operación por pulsos, las bobinas magnéticas se energizan por medio de una onda cuadrada de baja frecuencia. En un diseño específico se hace uso de una onda cuadrada unidireccional; esto es, un ciclo encendido-apagado de voltaje de CD. Durante el periodo de encendido de la excitación de la bobina electromagnética, la salida del flujómetro representa la señal más ruido; durante el periodo de apagado, la salida representa sólo el ruido. Restando la salida del periodo de apagado a la salida del periodo de encendido se obtiene una salida que consiste en la señal dependiente del flujo con un mínimo de contaminación por ruido. A flujo cero, esta sustracción elimina por completo el registro restante y se obtiene una salida cero exacta.

El flujómetro magnético no presenta obstáculos y es capaz de manejar fluidos y suspensiones sucios, corrosivos y de otras formas difíciles. Los únicos elementos en contacto directo con el fluido son la pared del tubo y los electrodos. En algunos diseños se incorpora una limpieza ultrasónica integrada de los electrodos.

La sección del tubo en sí debe fabricarse con material no magnético, excepto en algunos diseños en los que las bobinas magnéticas se encuentran dentro del tubo, que entonces forma una trayectoria magnética de regreso. Se requiere un forro aislante para el tubo, con excepción de las ocasiones en que se utilice para el tubo fibra de vidrio u otro material no conductor.

2.1.2.3. FLUJÓMETROS TÉRMICOS.

Existen dos tipos de principios para la detección de flujo en los que se emplea calor. El flujómetro anemómetro detecta las pérdidas de calor de un termómetro eléctrico de resistencia autocalentado debido al flujo del fluido.

En el anemómetro de corriente constante, un resistor que porta corriente inmerso en un fluido alcanza una temperatura determinada por el equilibrio de generación de calor $i^2 \cdot R$ y la pérdida de calor por convección. La resistencia del resistor sensible a la temperatura es de este modo una medida de la velocidad de flujo. Las resistencias utilizadas son alambres delgados (anemómetro de alambre caliente), películas delgadas y termistores, aunque también se han empleado circuitos en los que se utilizan termopares.

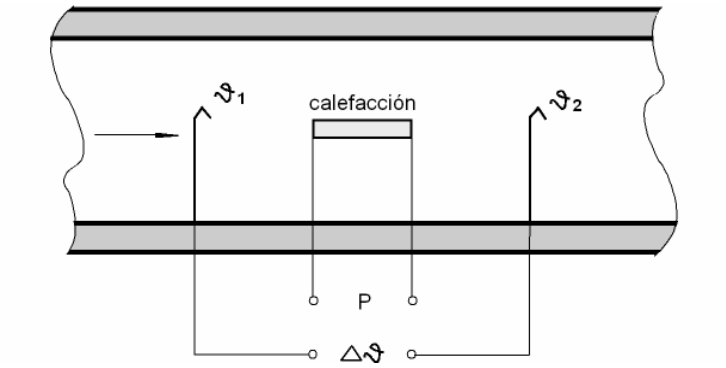


Figura 2.1.8 Flujómetro Térmico. [13]

Otro dispositivo, como el presentado en la figura 2.1.8, se denomina flujómetro térmico, consta de un calentador eléctrico inmerso en el fluido con dos detectores de temperatura equidistantes, uno corriente arriba y otro corriente abajo. La temperatura diferencial es función del gasto de masa. El calentador y los detectores de temperatura también pueden montarse fuera del tubo (flujómetro de capa límite); tal flujómetro tiende al caso ideal de medición no invasora. La figura anterior presenta un esquema de este tipo de flujómetro.

2.1.2.4. FLUJÓMETROS ULTRASÓNICOS.

La interacción de ondas sonoras con un fluido en movimiento puede emplearse para medir el gasto. Un flujómetro ultrasónico es un sistema de instrumentación que consta de uno o más transductores electroacústicos que operan en el intervalo de frecuencia de los Megahertz y unos circuitos electrónicos a fin de obtener la información del gasto.

El atractivo de tal sistema es que no es obstructivo y que existe el potencial para una medición verdaderamente no invasora, con los detectores de fluido situados fuera del tubo (flujómetro de abrazadera). Estas ventajas han motivado bastantes investigaciones y actualmente existen varios tipos de flujómetros ultrasónicos con exactitudes del orden del 0.5 % de la escala total de velocidades de flujo de 1 m/s o más sobre todo el intervalo.

En el medidor de diferencia de tiempo de tránsito se emplean dos transductores localizados relativamente corriente arriba y corriente abajo, en lados opuestos del tubo, ver figura 2.1.9. Se emiten ondas acústicas corriente arriba y corriente abajo, en una sucesión alternada, y se mide el tiempo de tránsito:

$$t_d = \frac{L}{c + V \cdot Cos\theta} \qquad \text{Ecuación 2.4}$$

$$t_u = \frac{L}{c - V \cdot Cos\theta} \qquad \text{Ecuación 2.5}$$

Donde t_d = tiempo de tránsito de la onda sonora corriente abajo.

t_u = tiempo de tránsito de la onda sonora corriente arriba.

c = velocidad acústica (velocidad del sonido en el fluido).

V = velocidad media de flujo del fluido.

L = distancia entre los transductores.

θ = ángulo entre la trayectoria sónica y la dirección de flujo del fluido.

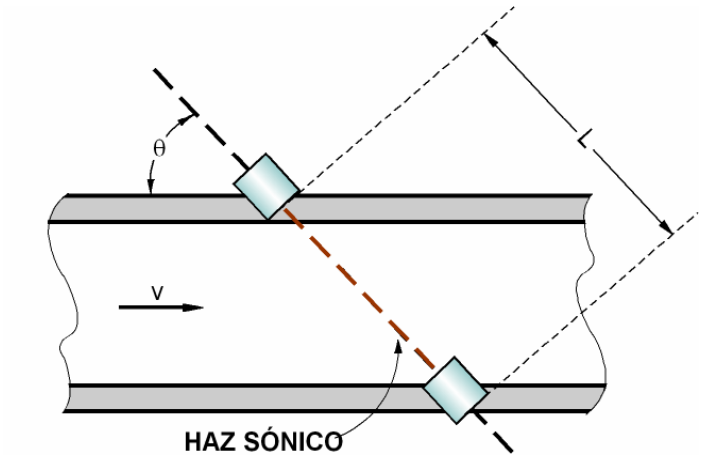


Figura 2.1.9 Flujómetro Ultrasónico. [11]

La diferencia de los inversos multiplicativos de los tiempos de tránsito determina una diferencia de frecuencia Δf:

$$\Delta f = \frac{1}{t_d} - \frac{1}{t_u} = \frac{2 \cdot V \cdot Cos\theta}{L} \qquad \text{Ecuación 2.6}$$

Proporcional a la velocidad del fluido e independiente de la velocidad del sonido en el fluido, y por consiguiente tampoco afectada por parámetros como la temperatura y composición, que afectan la velocidad del sonido. Este rendimiento ideal es aproximado en la práctica.

El flujómetro ultrasónico Doppler se basa en el principio de que la frecuencia de las ondas ultrasónicas reflejadas por partículas dispersas en el fluido en movimiento es desplazada en proporción a la velocidad de los dispersores, que pueden ser partículas sólidas suspendidas en el fluido o pequeñas burbujas de gas; pueden formar parte inherente del fluido por medir o bien introducirse deliberadamente con fines de medición ultrasónica Doppler.

Se supone que los dispersores se desplazan a la misma velocidad que el fluido mismo. Los flujómetros Doppler requieren un solo transductor, que puede ser humedecible (interior a la sección de medición) o de abrazadera (externo). Esta

última versión puede ser permanente, con una resina epóxica adhesiva, o portátil, con el transductor sujeto con masilla acústica o con grasa de silicón.

2.1.3. CIRCUITOS DE ACONDICIONAMIENTO.

No se puede hablar de los sensores, como componentes electrónicos aislados, sin ver como se pueden adaptar a un sistema de medición, supervisión, o control de procesos. Los circuitos completos de adquisición de datos, están separados en varios bloque funcionales que en la mayoría de los casos utilizan una topología como la mostrada en la figura 2.1.10.

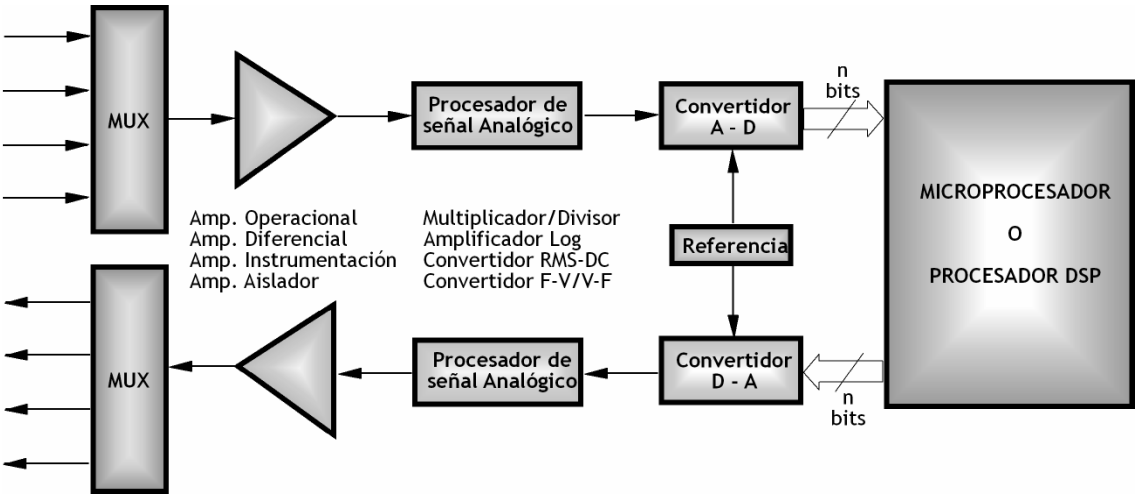


Figura 2.1.10 Esquema general de acondicionamiento y procesamiento de señal. [12]

Algunas de las tareas realizadas por el sistema total, son las siguientes:

- Generación de voltaje y frecuencia de excitación o de referencia.
- Generación de la señal de salida, típicamente por medio de un circuito puente.
- Acondicionamiento de la señal; es decir, amplificación de salidas de bajo nivel y ajuste de los valores de voltaje (o corriente) de salida a un intervalo estándar.
- Supresión de ruido, filtrado y aislamiento de tierra.
- Conversión de señales, tales como de CA/DC, o de analógico a digital.

- Procesamiento de señales, tales como linealización de salidas inherentemente no lineales.

Clásicamente algunos de los bloques descritos en la figura 2.1.10, inclusive los conversor A/D y D/A, se puede diseñar y construir utilizando amplificadores operacionales, que dependiendo de la calidad de la adquisición y manipulación de las señales requeridas, éstos podrían ser desde un amplificador operacional típico hasta uno especializado en cuanto al mejoramiento de sus características eléctricas.

Cuando el sistema de acondicionamiento consta de varias etapas separadas, las interconexiones forman parte del sistema total de medición. A fin de lograr el rendimiento especificado son esenciales la conexión correcta, el blindaje y la puesta a tierra.

Uno de los bloques principales y que debe rigurosamente, estar presente en todo sistema que involucre la medición de alguna variable física, es la etapa de acondicionamiento, la cual se encarga de preparar la señal proveniente del circuito de salida del sensor hacia las etapas posteriores, tomando en cuenta parámetros tales como la nivelación de voltajes, el acoplamiento de impedancias entre las etapas y el rechazo a las componente de ruido de la señal proveniente del sensor. El diagrama típico de este bloque se ilustra en la figura 2.1.11, y comúnmente se le llama circuito amplificador de instrumentación.

El amplificador de instrumentación convencional de tres amplificadores operacionales consta de dos etapas. La etapa de salida es un amplificador diferencial con la ganancia determinada por la relación característica de R_1 y R_2, R_4 es ajustable para compensar al máximo el CMRR de corriente continua. La red R_5 - C a veces se incluye para permitir ajustes para un voltaje de salida mínimo cuando se aplica una señal de modo común de 10 kHz y 2Vpp. Cada entrada del amplificador diferencial tiene un amplificador buffer, no inversor con retroalimentación negativa grande, por lo cual se mantiene la alta resistencia de entrada. La resistencia ajustable, R_A da un ajuste fino de la ganancia de las etapas del buffer y por ello controla con gran precisión la

ganancia total del amplificador. Para montajes prácticos, es recomendable utilizar amplificadores operacionales de bajo ruido.

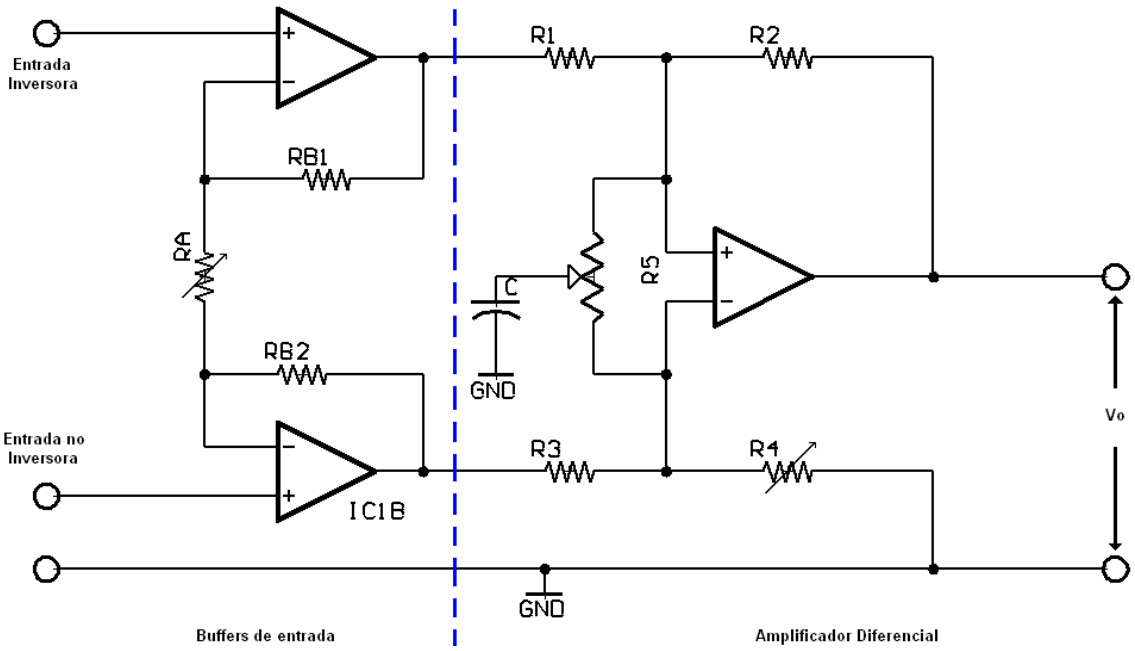


Figura 2.1.11 Amplificador típico de instrumentación. [10]

Gracias a los altos niveles de integración, los elementos electrónicos pueden estar integrados, localizados completamente en el interior del empaque del transductor. También en la actualidad es común encontrar en el mercado dispositivos electrónicos integrados, que cuentan internamente con algunas de las etapas del proceso y que con unos cuantos elementos externos, incluyendo el sensor, proporcionan una señal acondicionada y filtrada lista para ser procesada analógica o digitalmente. La desventaja de este tipo de dispositivo es que no poseen la capacidad de adecuarse a cualquier tipo de sensor por lo que debe ser utilizado con un único, o un grupo selecto, de dispositivos.

2.1.4. ACTUADORES.

Tal como se menciono anteriormente, el actuador es un transductor que responde a una señal eléctrica, presentando una salida con otro tipo de energía. Por conveniencia, los transductores de salida se clasifican en dos categorías principales:

- Actuadores electromagnéticos: Bajo este nombre se agrupan todos aquellos transductores de salida que aprovechando el campo magnético producido por electroimanes y amparados bajo las ecuaciones de Maxwell producen movimiento translacional o angular (motores y solenoides), control del flujo de líquidos (electroválvulas), conductividad eléctrica (relevadores) o sonidos audibles (altavoz o timbre de campana).

- Actuadores Electroluminicentes: La mayoría de estos actuadores, a excepción de la lámpara incandescente y el tubo de rayos catódicos, están fabricados de material semiconductor especial que irradia en forma de fotones la energía proveniente de la recombinación de los electrones con los huecos. Algunas aplicaciones de estos son la señalización de procesos (led's y lámparas incandescentes), transmisión de información inalámbrica (emisores infrarrojos), presentaciones alfanuméricas (displays de 7 segmentos, pantallas de rayos catódicos y pantallas de cristal líquido o LCD).

2.2. MICROCONTROLADORES.

El alto nivel de integración que se alcanzo entre la década de los 60's y 70's en la fabricación de dispositivos semiconductores, dio paso en un inicio, a la aparición del microprocesador y con ellos, una revolución en áreas tales como el procesamiento de información, control electrónico industrial, telecomunicaciones, etc.

Muy pronto, estos sistemas basados en microprocesadores eran demasiado grandes y complejos como para estar ubicados en equipos electrónicos dedicados a gobernar una sola

tarea. Bajo esa perspectiva y gracias al aumento de las técnicas de integración aparecen los microcontroladores, que básicamente son una pequeña computadora alojada dentro de un circuito integrado, con capacidades de procesamiento limitadas, pero de gran versatilidad para el desarrollo de aplicaciones, además con poco espacio por ocupar y de costo relativamente bajo.

Hoy en día es común encontrar microcontroladores en equipos electrónicos que se dedican al control de tareas que no requieren niveles altos de procesamiento o en aquellos que como limitante de diseño cuentan con poco espacio.

2.2.1. ¿QUÉ ES UN MICROCONTROLADOR?

Un microcontrolador es un circuito integrado programable que contiene todos los componentes de un computador, es decir, CPU, Memoria y Unidades de E/S. Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es la que le confiere la denominación de embedded controller.

El microcontrolador es un computador dedicado. En su memoria solo reside un programa destinado a gobernar una aplicación determinada; sus líneas de entrada/salida soportan la conexión de sensores y actuadotes del dispositivo a controlar y todos los recursos complementarios disponibles tienen como única finalidad atender sus requerimientos. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada.

2.2.1.1. DIFERENCIAS ENTRE LOS MICROPROCESADORES Y LOS MICROCONTROLADORES.

El microprocesador es un circuito integrado que contiene el CPU o procesador de una computadora. El CPU esta formado por la unidad de control, que interpreta las instrucciones, y el camino de datos, que las ejecuta.

Los pines de un microprocesador proporcionan al exterior las líneas de sus buses de direcciones, datos y control, para permitir la interacción con las memorias y los módulos de E/S y configurar un computador implementado por varios circuitos integrados. Se dice que un microprocesador es un sistema abierto porque su configuración es variable de acuerdo con la aplicación a la que se destine.

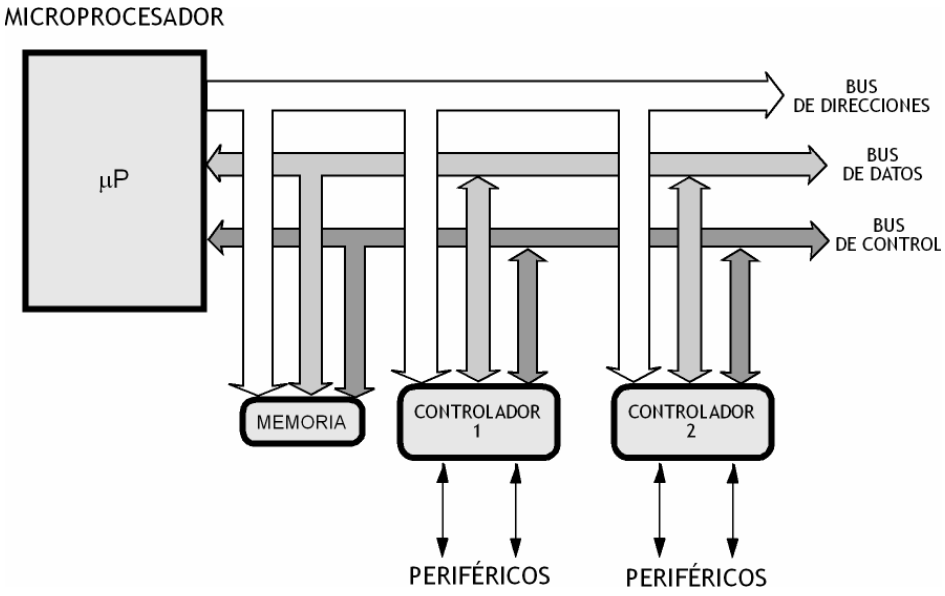


Figura 2.2.1 Sistema abierto basado en un microprocesador. [14]

Por el contrario, el microcontrolador únicamente pone a disposición del diseñador los pines necesarios para el control de los periféricos, formando un sistema cerrado en donde no es posible agregar periféricos externos.

Si solo se dispusiera de un modelo de microcontrolador, éste debería tener muy potenciados todos sus recursos para poder adaptarse a las exigencias de las

diferentes aplicaciones. Esta potenciación supondría en muchos casos un despilfarro. En la práctica cada fabricante oferta un elevado número de modelos diferentes, desde los más básicos hasta aquellos que cuentan con procesadores poderosos y recursos auxiliares elevados. Por ello, un aspecto muy destacado del diseño es la selección del microcontrolador a utilizar.

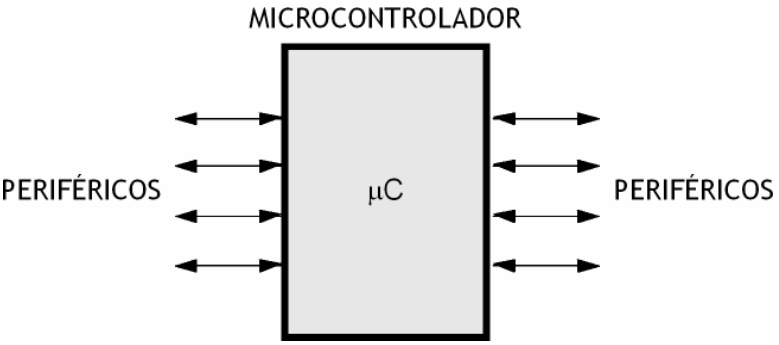


Figura 2.2.2 Sistema cerrado basado en un microcontrolador. [14]

2.2.2. ARQUITECTURA INTERNA.

Un microcontrolador posee todos los componentes de un computador, pero con unas características fijas que no pueden alterarse. Las partes principales de un microcontrolador son:

- Procesador
- Memoria no volátil para contener el programa.
- Memoria de lectura y escritura para guardar los datos.
- Líneas de E/S digitales.
- Líneas de E/S para los controles de periféricos:
  - Comunicación paralelo.
  - Comunicación serie.
  - Diversas puertas de comunicación (bus I^2C, USB, etc)
- Recursos auxiliares:
  - Circuito de reloj.
  - Temporizadores.

- Perro Guardián “Watchdog”
- Conversores AD y DA.
- Comparadores analógicos.
- Protección ante fallos de la alimentación.
- Estado de reposo o de bajo consumo.

2.2.2.1. EL PROCESADOR.

La necesidad de conseguir elevados rendimientos en el procesamiento de las instrucciones ha desembocado en el empleo generalizado de procesadores con arquitectura Harvard frente a los tradicionales que seguían la arquitectura Von Neumann. Esta última se caracteriza porque el CPU se conecta con una memoria única, en donde coexisten datos e instrucciones, utilizando un único sistema de bus. En la figura 2.2.3 se representa la forma básica de un procesador basado en arquitectura Von Neumann.

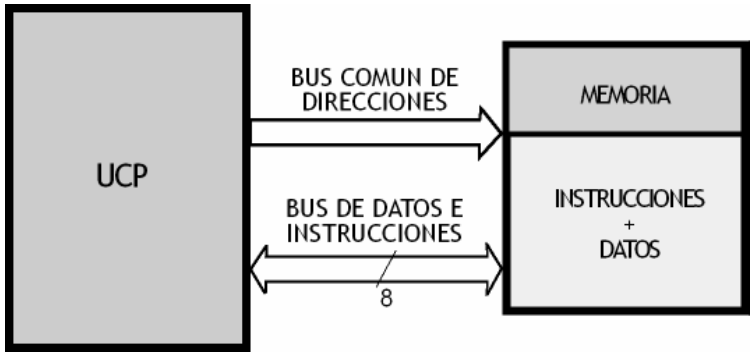


Figura 2.2.3 Arquitectura von Neumann. [14]

En la arquitectura Harvard son independientes la memoria de instrucciones y la memoria de datos y cada una dispone de su propio sistema de buses para el acceso. Esta dualidad, además de propiciar el paralelismo, permite la adecuación del tamaño de las palabras y los buses a los requerimientos específicos de las instrucciones y de los datos. Cabe mencionar también que la capacidad de memoria es diferente.

El procesador de los microcontroladores responde al tipo RISC (Reduced Instruction Set Computing), que se caracteriza por tener un repertorio de instrucciones maquina pequeño y simple, de forma que la mayor parte de las instrucciones se ejecutan en un ciclo de instrucción.

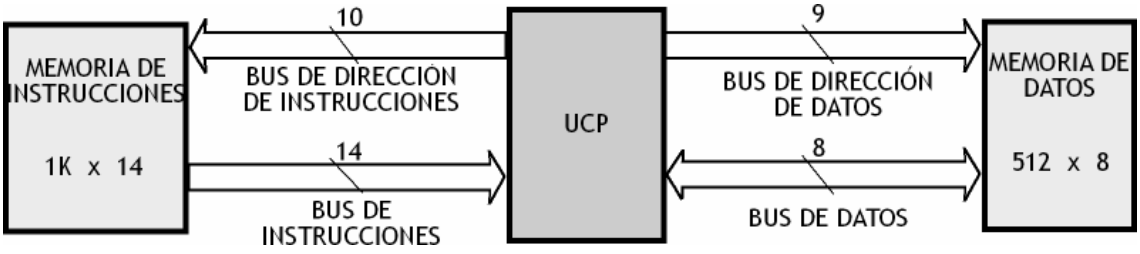


Figura 2.2.4 Arquitectura Harvard. [14]

Otra aportación frecuente que aumenta el rendimiento del computador es el fomento del paralelismo implícito, que consiste en la segmentación del procesados (pipe line), descomponiéndolo en etapas para poder procesar una instrucción diferente en cada una de ellas y trabajar con varias a la vez.

2.2.2.2. MEMORIA DE PROGRAMA.

El microcontrolador esta diseñado para que en su memoria de programa se almacene todas las instrucciones del programa de control. No hay posibilidades de utilizar memorias externas de ampliación.

Como el programa a ejecutar siempre es el mismo, debe estar grabado de forma permanente. Los tipos de memoria adecuados para soportar esta función admiten cinco versiones diferentes:

- ROM con mascara: En este tipo de memoria el programa se graba en el chip durante el proceso de su fabricación mediante el uso de máscaras. Los altos

costos de diseño e instrumental solo aconsejan usar este tipo de memoria cuando se precisan serie muy grandes.

- EPROM: La grabación de esta memoria se realiza mediante un dispositivo físico gobernado desde un computador personal, que recibe el nombre de quemador. En la superficie de cada cápsula del microcontrolador existe una ventana de cristal por la que se puede someter al chip de la memoria a radiaciones ultravioletas para producir su borrado y emplearla nuevamente. Es una ventaja contar con este tipo de memoria en la fase de desarrollo de alguna aplicación, pero su costo unitario es elevado.

- OTP (Programable una vez): Este modelo de memoria sólo se puede grabar una vez por parte del diseñador, utilizando el mismo procedimiento que con la memoria EPROM. Posteriormente no se puede borrar. Su bajo precio y la sencillez de las grabaciones predisponen el uso de este tipo de memoria para prototipos finales y series de producción corta.

- EEPROM: La grabación es similar a las memorias OTP y EEPROM, pero el borrado es mucho mas sencillo al poderse efectuar de la misma forma que el grabado, es decir, eléctricamente. Sobre el mismo zócalo del quemador puede ser programada y borrada tantas veces como se quiera, lo cual la hace ideal en la enseñanza y en la creación de nuevos proyectos. Aunque se garantizan $1_1000{,}000$ de ciclos de escritura/borrado en una EEPROM, todavía su tecnología de fabricación tiene obstáculos para alcanzar capacidades importantes y el tiempo de escritura de las mismas es relativamente grande y con elevado consumo de energía.

- FLASH: Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar en circuito al igual que las EEPROM, pero suelen disponer de mayor capacidad que estas ultimas. El borrado solo es posible con bloques completos y no únicamente sobre alguna localidad específica. Son muy

recomendadas en aplicaciones en las que sea necesario modificar el programa a lo largo de la vida del producto.

2.2.2.3. MEMORIA DE DATOS.

Los datos que manejan los programas varían continuamente, y esto exige que la memoria que les contiene deba ser de lectura y escritura, por lo que la memoria RAM estática (SRAM) es la más adecuada, aunque sea volátil.

Existen microcontroladores que disponen como memoria de datos una de lectura y escritura no volátil, del tipo EEPROM. De esta forma, la ausencia del suministro energético no ocasiona la perdida de la información, que está disponible al reiniciarse el programa.

2.2.2.4. LINEAS DE E/S PARA EL CONTROL DE PERIFERICOS.

A excepción de dos pines destinados a recibir la alimentación, otras dos para la señal de reloj proveniente de un cristal y una mas para provocar el Reset, los pines restantes de un microcontrolador sirven para soportar su comunicación con los periféricos externos que controla.

Las líneas de E/S que se adaptan con los periféricos manejan información en paralelo y se agrupan en conjuntos de ocho, que reciben el nombre de puertos. Hay modelos con líneas que soportan la comunicación en serie; otros disponen de conjuntos de líneas que implementan puertos de comunicación para diversos protocolos, como I^2C, USB, IrDA, entre otros.

2.2.2.5. RECURSOS AUXILIARES.

Según las aplicaciones a las que oriente el fabricante cada modelo de microcontrolador, incorpora una diversidad de complementos que refuerzan la potencia y la flexibilidad del dispositivo. Entre los recursos mas comunes se citan los siguientes:

- Circuito de Reloj: Encargado de generar los impulsos que sincronizan el funcionamiento de todo el sistema.
- Temporizadores: Orientados a controlar tiempos.
- Perro Guardián: Destinado a provocar una reinicialización cuando el programa queda bloqueado.
- Conversores AD y DA: Para la comunicación con elementos externos analógicos.
- Comparadores analógicos: Para verificar el valor de una señal analógica.
- Sistema de protección ante fallos de la alimentación.
- Estado de Reposo, en el que el sistema queda congelado y el consumo de energía se reduce al mínimo.

2.2.3. PROGRAMACION DE MICROCONTROLADORES.

La utilización de los lenguajes mas cercanos a la maquina (de bajo nivel) representan un considerable ahorro de código en la confección de los programas, lo que es muy importante dada la estricta limitación de la capacidad de la memoria de instrucciones. Los programas bien realizados en lenguaje Ensamblador optimizan el tamaño de la memoria que ocupa y su ejecución es muy rápida.

Los lenguajes de alto nivel más empleados con microcontroladores son el C y BASIC, de los que existen varias empresas que comercializan versiones de compiladores e intérpretes para diversas familias de microcontroladores.

Hay versiones de intérpretes de BASIC que permiten la ejecución del programa línea a línea y, en ocasiones, residen en la memoria del propio microcontrolador. Con ello se puede escribir una parte del código, ejecutarlo y comprobar el resultado antes de proseguir.

2.2.4. HERRAMIENTAS DE DESARROLLO.

Siempre que se diseñan aplicaciones con circuitos integrados programables se precisan herramientas para la puesta a punto del hardware y del software.

Con respecto al software, además de los compiladores o intérpretes de los lenguajes usados, es muy importante disponer de simuladores software, que consisten en programas que simulan la ejecución de instrucciones representando el comportamiento interno del procesador y el estado de las líneas E/S. Como se simula por software al procesador, el comportamiento no es idéntico aunque proporciona una aproximación aceptable, especialmente cuando no es esencial el trabajo en tiempo real.

Respecto a las herramientas hardware, una indispensable es el quemador, encargado de escribir el programa en la memoria del microcontrolador. Existen quemadores muy completos capaces de trabajar con muchos modelos de diferentes familias, pero su elevado precio los aleja de los pequeños desarrolladores. Para estos últimos existen bastantes versiones de sencillos grabadores, específicos para ciertos modelos de microcontroladores, que gobernados desde un computador personal se ofrecen por un precio accesible.

2.3. INTERFACES DE COMUNICACIÓN ENTRE DISPOSITIVOS ELECTRÓNICOS.

El término interfaz es de uso muy frecuente en informática. Es un concepto abstracto, que puede referirse a un dispositivo físico (hardware) o a un dispositivo lógico (software),

aunque lo común es que se trate de una mezcla de ambos. Se refiere al ámbito de conectividad local entre dos sistemas.

Para poder establecer una comunicación entre dos dispositivos electrónicos o sistemas, con la finalidad de intercambiar datos entre ellos, es necesario contar con un método de comunicación congruente y valido entre ambos terminales. Con tal objetivo, fabricantes del sector electrónico han desarrollado interfaces en conjunto con organismos reguladores, cumpliendo con ciertas normas de estandarización con el fin de lograr total conectividad entre los sistemas, es decir, estas interfaces se basan en el modelo de referencia OSI[23]. Algunos ejemplos de este tipo son IEEE 1284 (Parallel Port Printer), EIA RS232 (Serial Communications), IEEE 1394 (FireWire) o IEEE 802.15 (WPAN/Bluetooth) por citar algunos ejemplos.

Por otro lado, existen interfaces propietarias "abiertas", tales como USB (Universal Serial Bus) o IrDA (Infrared Data Association), en donde el grupo o foro desarrollador publica las especificaciones eléctricas, mecánicas y lógicas. Otro tipo de interfaces son las propietarias cerradas, que por razones como la seguridad, la propiedad intelectual o por estrategias mercadológicas; las especificaciones no son divulgadas.

En los apartados siguientes, se realiza una descripción breve sobre algunas interfaces de comunicación más usuales, encontradas en dispositivos móviles, computadoras o sistemas de adquisición de datos.

2.3.1. ESTÁNDAR RS232.

El estándar RS232-C propuesto por la EIA (American Electronic Industries Association), es uno de los estándares de mayor aceptación internacional en la transferencia de datos. El CCITT europeo (Internacional Telegraph and Telephone Consultive Comittee) ha emitido un estándar homologo con el número V.24.

[23] OSI: Open Systems Interconection.

En general la comunicación definida en el estándar RS232 es de tipo serie y permite la transmisión síncrona y asíncrona; siendo la normativa asíncrona la más difundido en los equipos electrónicos.

Esta interfaz, fue concebida inicialmente para establecer una comunicación entre dos hosts, interconectados a través de una línea telefónica auxiliados por módems. En la actualidad, además del uso mencionado anteriormente, y a pesar de la existencia de otras interfaces con mayor valor agregado, RS232 continua siendo utilizada por dispositivos periféricos y otros equipos electrónicos, como vía de acceso para efectuar tareas de sincronización de datos, programación de funciones, cambios en configuraciones de operación o ejecución de diagnósticos.

En el entorno industrial el peso de RS232 es también muy importante. Si bien existen soluciones de comunicación serie más robustas y versátiles, como RS422 o RS475, RS232 sigue siendo por su sencillez, su diseño económico y, sobre todo, por su gran difusión, la norma más frecuente.

2.3.1.1. UART: FUNDAMENTO DE LA COMUNICACIÓN SERIE.

El dispositivo clave de un sistema de comunicaciones serie, es el periférico llamado UART (Universal Asynchronous Receiver Transmitter), siendo su función principal la conversión de datos paralelos a serie, durante la transmisión y viceversa mientras el sistema establece la recepción de información.

En la figura 2.3.1, se muestra el esquema general con los bloques básicos de un controlador UART. Se distinguen los registros de datos, tanto de recepción como de transmisión y sus correspondientes registros de desplazamiento (RxD, TxD), los registros de control de transmisión y recepción y las líneas que transportan las señales de sincronización para comienzos la transmisión y recepción (RTS, CTS)

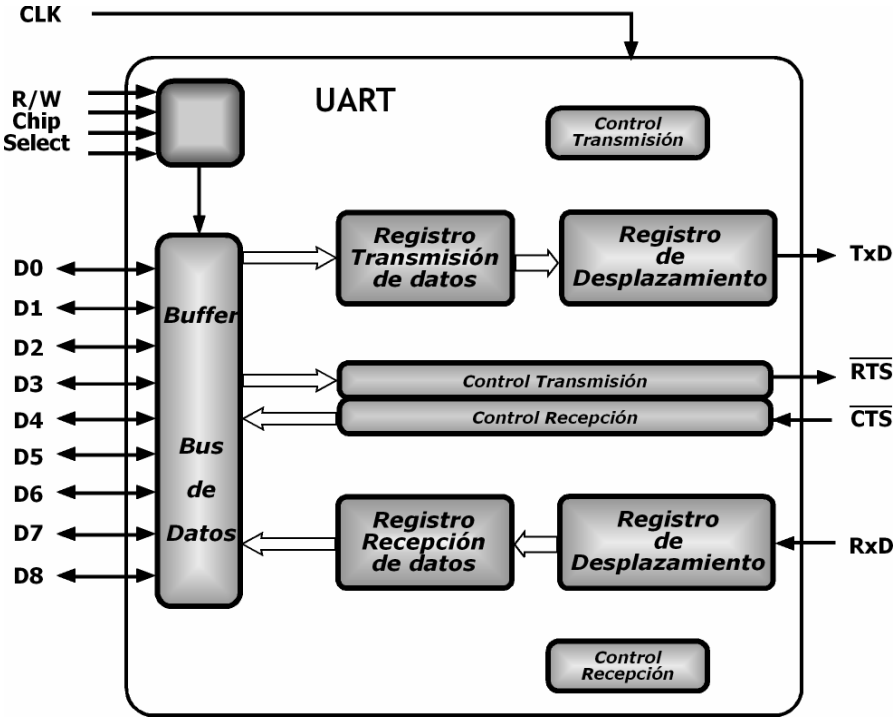


Figura 2.3.1 Estructura Interna del controlador UART [15]

Gracias a la programabilidad del dispositivo, es posible mediante software configurar la velocidad de transferencia, el tamaño en bits de la palabra de datos y habilitar la revisión y el tipo de paridad, entre algunas otras funciones.

El controlador UART se interconecta con otros dispositivos electrónicos que acondicionan los niveles de voltaje para poder cumplir con las recomendaciones especificadas en EIA RS232.

2.3.1.1.1. TRANSMISIÓN SERIA ASINCRÓNA.

La comunicación asíncrona, permiten que los datos sean transmitidos sin necesidad que el emisor envié una señal de reloj para lograr una sincronización con el sistema receptor. Por lo tanto, el transmisor y el receptor manejan la transferencia de datos a través de la inserción de un par de bits especiales a cada palabra que se pretende transmitir.

Cuando un byte o palabra está lista para ser transmitida, un bit llamado Bit de Inicio (Start Bit) es añadido a la trama y con ello se logra indicarle al receptor que los bits subsecuentes pertenecen a un dato valido, y también forzar a que el reloj del receptor entre en sincronización con el reloj del transmisor.

Después de la recepción del bit de inicio, cada uno de los bits que componen la palabra de datos es enviado uno a uno, transmitiendo inicialmente el bit menos significativo (LSB). Luego de completar todos los bits del campo de datos y si está activa la revisión de paridad, el transmisor agrega un Bit de Paridad (Parity Bit) que es utilizado por el receptor como un método sencillo para la detección de errores. Inmediatamente después, se cierra la trama de datos con el envío del Bit de Paro (Stop Bit).La figura 2.3.2, muestra una trama típica de UART, con ocho bits de datos, sin bit de paridad y un bit de parada.

Si el bit de paro no se detecta en el receptor, cuando se supone que debe estar presente, el controlador UART considera que la palabra completa ha sido confusa y puede reportar un error de entramado hacia el procesador del host receptor cuando la palabra de datos es leída. Usualmente, la causa de errores de entramado, es producida por que ambos terminales están configurados a velocidades de transferencias diferentes; o por que la señal sufrió algún tipo de interrupción en el medio físico.

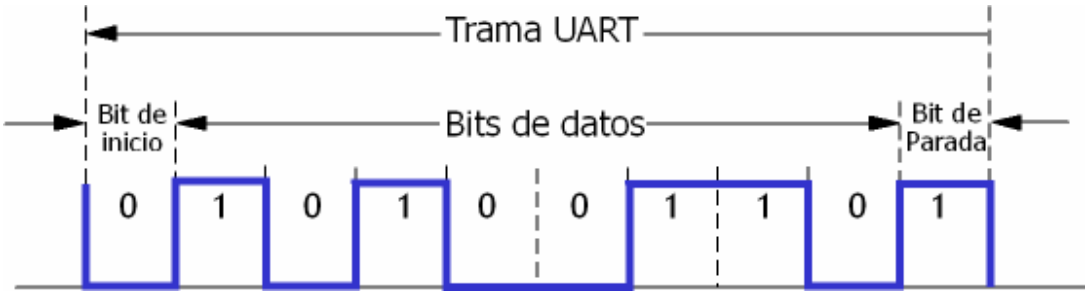


Figura 2.3.2 Trama típica 8N1 (8 bits de datos, ningún bit de paridad y un bit de paro) de comunicación serie asíncrona. [15]

A pesar de que los datos se hayan recibido correctamente o no, y si ambos terminales poseen la misma configuración, los bits de Inicio, Paridad y Paro son descartados de forma automática y únicamente se entregan los bits de la palabra de datos como información recibida.

Si otra palabra esta lista para ser enviada, el nuevo bit de inicio puede ser transmitido a continuación del bit de paro del byte enviado previamente. Si no existen datos para transmitir, el canal físico permanece inactivo.

2.3.1.1.2. OTRAS FUNCIONES DE LA UART.

En adición al trabajo básico desarrollado por el dispositivo de control UART, éste, también suministra señales de Handshaking para interfaces RS232. Estas señales se utilizan para inicial la comunicación y regular el flujo de datos en el caso que el dispositivo remoto no este preparado para aceptar mas información.

2.3.1.2. SIMBOLIZACIÓN DE LOS BITS.

En RS232, el estado lógico alto, es llamado Marca (Mark) y el estado lógico bajo, es conocido como Espacio (Space). Cuando la línea de comunicación esta inactiva, se dice que esta "Marcando", o transmitiendo continuamente el estado lógico alto.

El bit de inicio se transmite como un Espacio y el bit de paro se considera una Marca. Esto significa que cada vez que se reconozca el cambio de una Marca a un Espacio (de estado lógico alto a un estado lógico bajo) es porque existirá una nueva palabra a ser transferida, aún en aquellos casos en donde se transmitan palabras continuas. Esto además garantiza que el transmisor y el receptor puedan resincronizar sus relojes a pesar de estar iniciando una nueva transmisión.

2.3.1.3. ESPECIFICACIONES DEL ESTÁNDAR.

2.3.1.3.1. ESPECIFICACIONES MECÁNICAS (ISO 2110).

Las especificaciones del estándar definen veinticinco hilos, terminados en un conector DB25 en sus versiones macho (Host) y hembra (Modem) para efectuar las tareas de transmisión, recepción y control de la comunicación (Handshaking). Sin embargo, la comunicación es posible con menos de veinticinco líneas de funciones, ocupando solo nueve de ellas (para comunicación asíncrona exclusivamente) y confinándolas en un conector tipo DB9, siguiendo la misma peculiaridad en cuanto al genero del conector, tal como en el DB25.

En la figura siguiente se muestran los conectores DB25 y DB9, utilizados para la comunicación serie con RS232. El estándar define todas las dimensiones y características que estos deben cumplir.

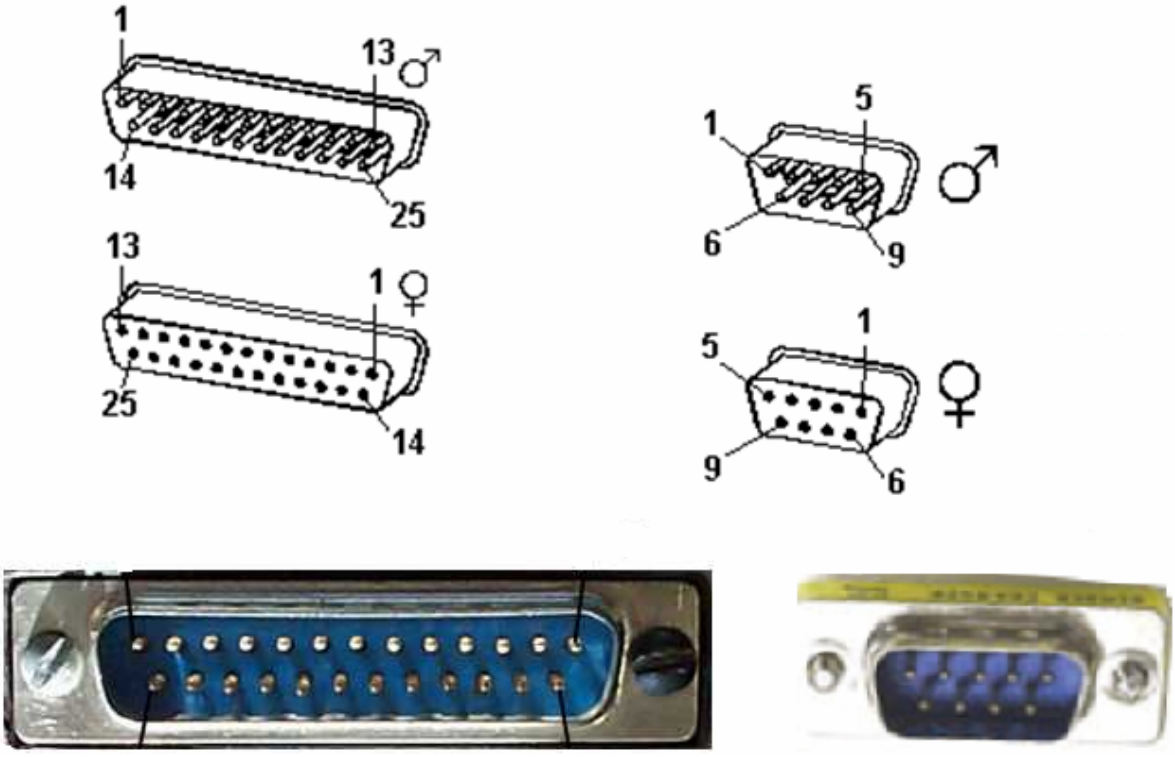


Figura 2.3.4 Conectores DB25 y DB9.

También se establece una longitud máxima aproximada de quince metros para unir un equipo terminal con un Modem. Esta longitud, también esta ligada a propiedades eléctricas tales como la velocidad de transmisión y la capacitancia del cable. Se considera como relación práctica, una disminución en la tasa de transferencia y de la calidad de la misma, si se aumenta la longitud del cable.

2.3.1.3.2. ESPECIFICACIONES ELÉCTRICAS.

La norma, fija una transmisión en modo común, cada circuito tienen una referencia a tierra y esta es común para todos los circuitos. Los circuitos son punto a punto, es decir, un driver con un sólo receptor de la señal.

También define el uso de una señal bipolar de lógica invertida conocido como NRZ-I[24], para codificar los bits en el canal de comunicación. Las amplitudes de esta señal son simétricas con respecto a tierra. Con ello se logra desvincular un potencial de cero voltios con los estados lógicos uno y cero.

Eléctricamente, una Marca se considera como tal si es un potencial válido dentro del rango de -3V a -15V; por otro lado un estado lógico bajo o Espacio, estará definido como aquel potencial que se encuentra entre +3V y +15V. La figura 2.3.5 presenta un ejemplo de este tipo de codificación.

Solamente cuatro de los veinticinco hilos se usan para funciones de datos, el resto están reservados para funciones de control, temporización, tierra y pruebas. Estos hilos adicionales tienen una especificación eléctrica similar a los de datos, se consideran el estado On si transmiten una tensión por encima de los +3V y Off si están por debajo de –3V. En la práctica, los niveles de tensión que se utilizan son los que proporcionan las fuentes de alimentación de ±12V.

[24] NRZ-I: Non Return to Zero Inverted

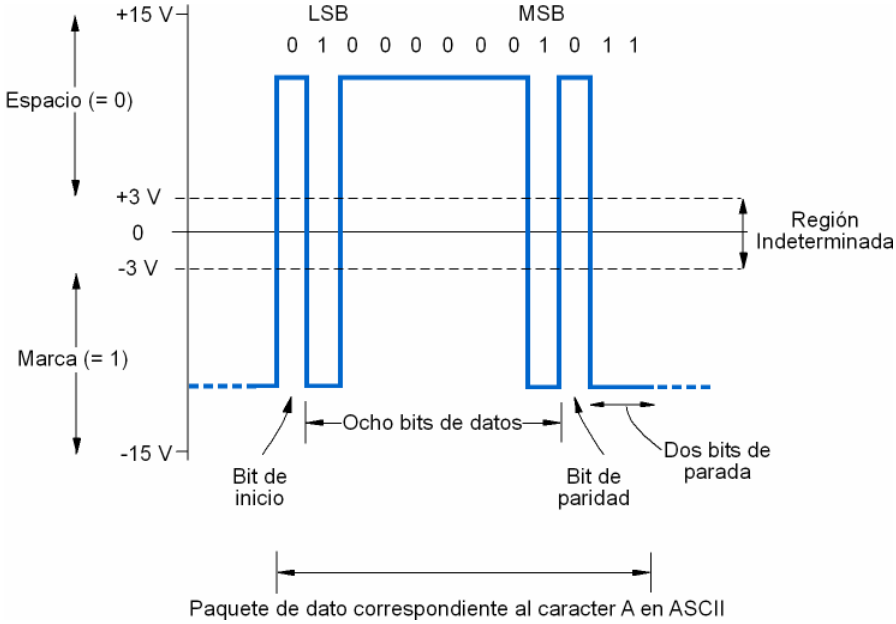


Figura 2.3.5 Codificación NRZI utilizada en el estándar RS232. [15]

En lo que respecta a la velocidad de transmisión, esta se encuentra ligada con el modelo del dispositivo UART. Originalmente la máxima velocidad alcanzable era de 20 kbps, hoy en día es posible obtener velocidades de hasta 115 kbps.

Otros aspectos eléctricos importantes mencionan que, al menos teóricamente, los drivers de salida de las puertas disponen de un mecanismo de auto-protección contra sobrecalentamientos, lo cual mantiene al sistema protegido de cortocircuitos. La tensión máxima de operación es ± 25 voltios y la carga máxima oscila entre 3 kΩ y 7 kΩ., con una corriente máxima de salida igual a 500 mA.

2.3.1.3.3. ESPECIFICACIONES FUNCIONALES.

Se presenta una tabla a continuación, con el nombre y funciones de cada línea para conectores DB25 y DB9:

DB25 (RS232-C)	DB9 (PC IBM)	NOMBRE	FUENTE DE LA SEÑAL	DESCRIPCIÓN
1	-	PG/FG	-	Frame/Protective Ground
2	3	TD	DTE	Transmit Data
3	2	RD	DCE	Receive Data
4	7	RTS	DTE	Request to Send
5	8	CTS	DCE	Clear to Send
6	6	DSR	DCE	Data Set Ready
7	5	SG/GND	-	Signal Ground
8	1	DCD/CD	DCE	Data Carrier Detect
9	-	-	-	Reserved for Test
10	-	-	-	Reserved for Test
11	-	-	-	Reserved for Test
12	-	SRLSD	DCE	Sec. Recv. Line Signal Detector
13	-	SCTS	DCE	Secondary Clear to Send
14	-	STD	DTE	Secondary Transmit Data
15	-	TSET	DCE	Trans. Signal Element Timing
16	-	SRD	DCE	Secondary Received Data
17	-	RSET	DCE	Receiver Signal Element Timing
18	-	LOOP	DTE	Local Loopback
19	-	SRS	DTE	Secondary Request to Send
20	4	DTR	DTE	Data Terminal Ready
21	-	RDL	DTE	Remote Digital Loopback
22	9	RI	DCE	Ring Indicador
23	-	DSRS	DTE	Data Signal Rate Selector
24	-	TSET	DTE	Trans. Signal Element Timing
25	-	-	DCE	Test Mode

Tabla 2.3.1 Resumen de Funciones de cada circuito de RS232. [15]

Las funciones descritas a continuación, son de aquellos pines o circuitos de mayor importancia y que concuerdan con los necesarios para la comunicación asíncrona (contenidos en un conector DB9). Estas nueve líneas se pueden separar en tres bloques funcionales.

- Bloque de establecimiento de conexión, lo forman las líneas:
  - RI (Ring Indicator): Indicador de llamada. Con esta señal el módem indica al host que se ha producido una llamada desde el módem remoto. Cuando se produce una llamada se activa (se pone a ON), y entre tonos o sin recepción de llamada esta señal esta desactivada (OFF).

  - DTR (Data Terminal Ready): Terminal de datos listo. Esta señal en estado ON indica al módem que el PC está conectado y listo para comunicar. Si la señal se pone a OFF mientras el módem está conectado, el módem termina la sesión y cuelga el teléfono.

  - DSR (Data Set Ready): Módem activo. Es un circuito de control. Si está ON el módem está preparado para intercambiar señales de control con la PC. Si está OFF, el módem no está preparado para aceptar señales de datos del host.

Este bloque tiene como objetivo indicarle a los PCs, que han establecido un canal de comunicación (normalmente a través de la línea telefónica). Las líneas DTR y DSR del equipo local y del remoto deben estar activas (set) durante todo el proceso. De hecho cuando un PC desea dar por terminada una conexión basta con que, momentáneamente, desactive su DTR.

La conexión se inicia manualmente (el usuario llama con el teléfono al modem remoto) o automáticamente (el modem tiene capacidad de marcar un número de teléfono) y se gestiona en los modems, que negocian de forma automática, los parámetros de transferencia como la velocidad, compresión, etc.

Se asume que el usuario del PC que llama activará el proceso que va a utilizar la conexión. En el PC llamado se asume también que el proceso homólogo está ya activo o se puede activar automáticamente al recibir de su modem la señal de RI. Sea como fuera, la conexión queda establecida. A partir de este momento los PCs pueden intercambiar información.

- Bloque de control de flujo. Estas líneas tienen sentido en el caso de que el canal de comunicación establecido tenga una gestión half-duplex. Si el canal está establecido, el protocolo software de nivel de enlace de datos que se esté utilizando (Xmodem, Ymodem, HDLC, etc.) fijará cuál de los dos hosts debe comenzar a hablar/transmitir. Las líneas en este bloque son usadas de la siguiente manera:

  - DCD (Carrier Detect): Detección de portadora. El módem indica al PC que está conectado con otro módem.

  - RTS (Request to Send): Petición de transmisión. Se pone a ON cuando el PC quiere enviar datos, a OFF cuando el PC no tiene nada que transmitir.

  - CTS (Clear to Send): Preparado para transmitir. Se pone a ON cuando el módem puede aceptar más datos del PC a OFF cuando no es posible por estar en esos momentos realizando otra operación.

El PC que quiere transmitir activa RTS, entonces el módem transmisor envía una señal portadora, sin modular, para avisar al módem remoto que se reserva el canal. Una vez reservado el canal el módem transmisor comunica a

su host que ya puede transmitir activando la línea CTS. Cuando un PC haya terminado de transmitir, desactivará RTS, el módem quitará la portadora y desactivará CTS. Entonces el otro módem podrá reservar el canal si su PC desea transmitir.

En caso de que la gestión del canal sea full-dúplex todo es más sencillo. Cuando un PC quiere transmitir activa su RTS. Automáticamente su modem le da paso activando CTS.

- Bloque de Transmisión/Recepción de datos, la transferencia serie de la trama UART, se produce en las líneas:
  - TX y RX (Transmisión y Recepción): Transmisión de datos y Recepción de datos.

2.3.1.4. DISPOSITIVOS DTE Y DCE.

Las especificaciones definen dos tipos de equipo: El equipo terminal de datos o DTE (Data Terminal Equipmet), y un equipo portador de datos, llamado DCE (Data Carrier Equipment).

Usualmente el dispositivo DTE es una computadora, y el DCE es un Modem. Cruzando la línea telefónica, el Modem Receptor también es un dispositivo DCE y la computadora receptora, conectada al Modem receptor, es un dispositivo DTE. La figura 2.3.6 presenta la conexión de dos hosts por medio de Modems.

La figura 2.3.7 muestra la conexión entre dos dispositivos DTE sin la intervención de un Modem, es necesario utilizar una configuración de cableado llamado Null Modem, la cual consiste en un cable cruzado que evita un cortocircuito en las líneas de salida de ambos dispositivos.

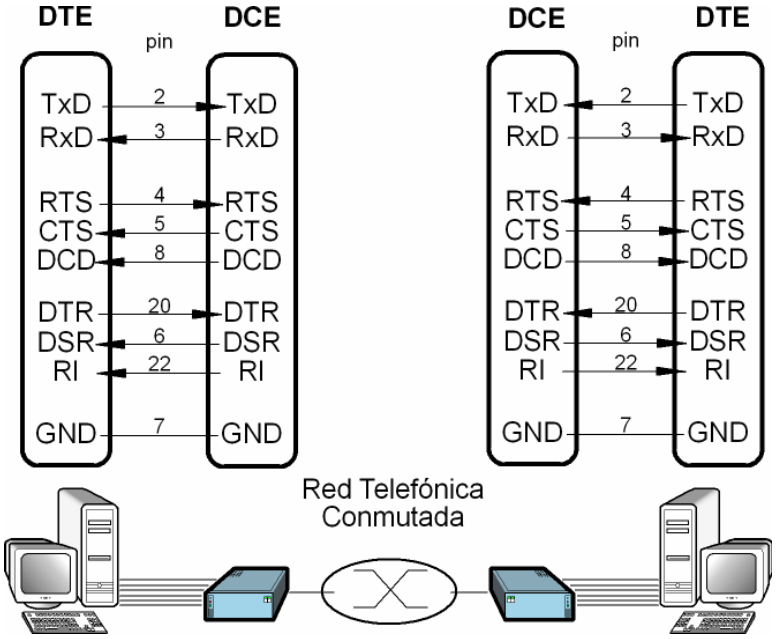


Figura 2.3.6 Conexión entre dos hosts utilizando modems. [15]

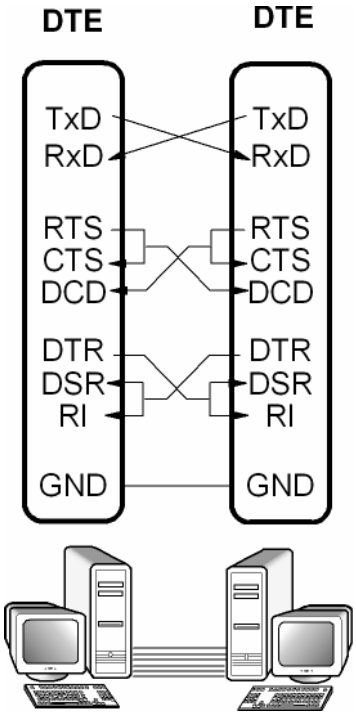


Figura 2.3.7 Conexión entre dos hosts sin modem. [15]

Para entender la lógica de este cableado, debe observarse en la figura 2.3.7, que hay dos líneas independientes de transmisión de datos, una en cada sentido. Por lo tanto la comunicación es potencialmente Full-dúplex. Esto implica que cada DTE puede transmitir cuando lo desee, independientemente de que el otro DTE lo esté haciendo o no.

Por lo tanto cuando un PC quiere transmitir activa su RTS, esto activa también su propia CTS lo que le permite transmitir inmediatamente. Además indica que está transmitiendo a la máquina remota activando el DCD remoto.

2.3.2. IRDA.

IrDA es la abreviación de Infrared Data Association, una organización sin fines de lucro que establece los estándares para entablar una comunicación tipo serie vía radiación infrarroja entre dos dispositivos.

IrDA nació en el año de 1,993 para establecer y dar respaldo a los estándares tanto a nivel físico como lógico con el objetivo de poder crear enlaces de comunicación infrarroja. La asociación decidió crear una interconexión serie de datos, de bajo costo, baja potencia, que transmita en modo half duplex, que soporte un modelo de conectividad punto a punto y con facilidad a adaptarse a un amplio rango de aplicaciones y dispositivos.

La comunicación infrarroja (IR), esta basada en una tecnología similar a la ocupada por los dispositivos de control remoto, usados en la operación de equipo electrónico doméstico. IR ofrece una forma conveniente, barata y confiable de conectar computadoras y dispositivos periféricos de manera inalámbrica.

IR es un enlace de datos interoperable que ofrece bajo consumo de energía, aun para transferencias con velocidades cercanas a los 4 Mbps, lo que permite establecer

una comunicación universal, que es utilizada por PC's, PDA's, Impresoras, teléfonos móviles, puntos de acceso a redes, entre otros.

2.3.2.1. ESTÁNDAR IRDA.

La versión actual del estándar IrDA es la 1.3, que permite un máximo de velocidad igual a 4 Mbps, aunque pronto se espera que aparezcan dispositivos que soporten la versión 1.4 con una tasa de transferencia de datos superior a los 16 Mbps.

2.3.2.1.1. ESPECIFICACIONES ELÉCTRICAS.

La transmisión en modo compatible (algunas veces llamado SIR, por Serial IR) utiliza en el caso mas simple, una comunicación directa con la UART incluida en la mayoría de PC's, y con una simple interfaz que recorta la duración de un bit hasta un máximo de 3/16 del tiempo original, como requerimiento para el ahorro energético. Un diodo emisor infrarrojo es manipulado eléctricamente para transmitir una señal óptica al receptor.

Las figuras 2.3.8 y 2.3.9 presentan el diagrama de bloques para la implementación de la comunicación infrarroja y los formatos de las tramas utilizadas por el controlador UART y la salida Infrarroja, respectivamente.

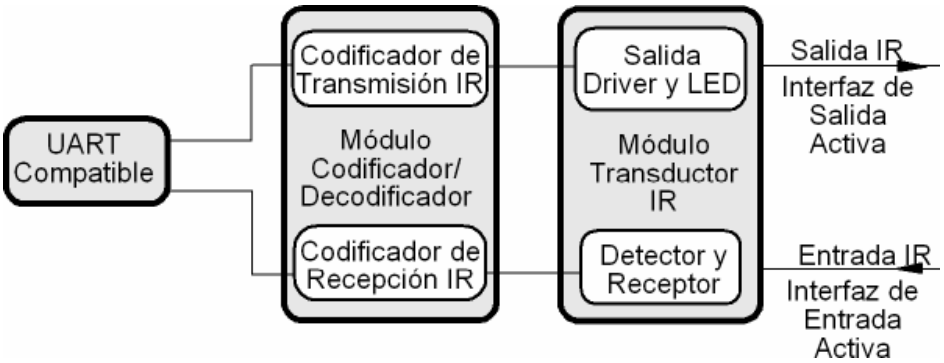


Figura 2.3.8 Diagrama de bloques del proceso de acondicionamiento para estableces comunicación infrarroja. [16]

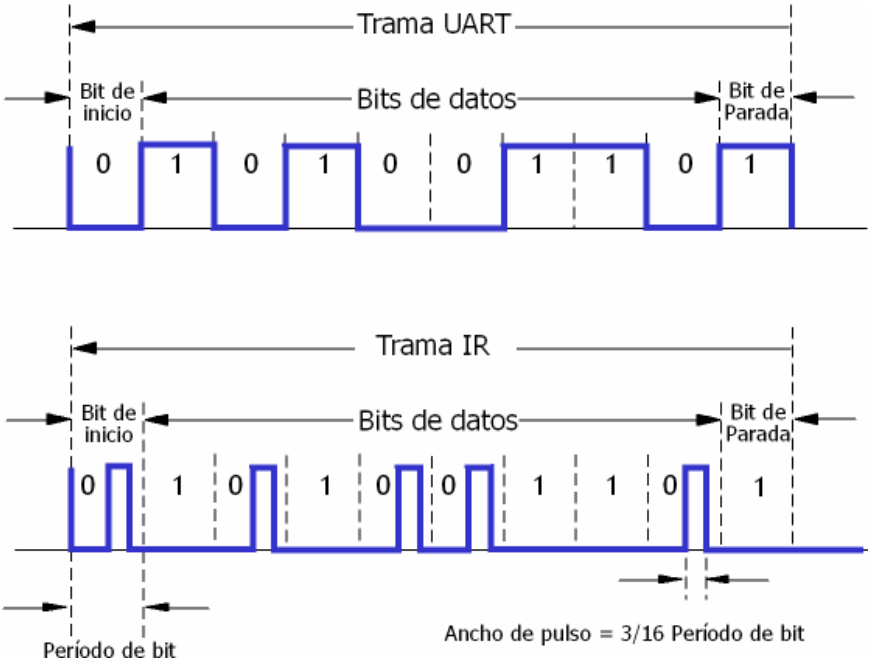


Figura 2.3.9 Comparación de las tramas de datos en UART e IrDA. [16]

Este modo de transmisión cubre hasta una velocidad máxima de 115.2 kbps, que corresponde también a la máxima velocidad soportada por los controladores UART estándar. La mínima velocidad de transmisión soportada es de 9.6 kbps. Todas las transmisiones deben ser iniciadas a ésta velocidad para establecer compatibilidad, el hecho de poder establecer velocidades mayores de transmisión, es el resultado de una negociación del software de control en ambas entidades, una vez establecido el enlace.

Para mayores velocidades, se requiere una interfaz especial, que opere a 1.152 Mbps y utilice un proceso de recorte similar al modo SIR, pero con una reducción de 1/4 de duración del pulso original. La máxima velocidad soportada, en la versión 1.3, es de 4.0 Mbps (Llamada FIR de Fast IR), que opera con pulsos de 125ns de duración utilizando un esquema de modulación 4-PPM[25].

[25] PPM: Pulse Position Modulation.

Para frecuencias superiores a los 115.2 kbps, la mínima intensidad de salida es de 40mW/sr. Para altas velocidades es requerida una intensidad de 100mW/sr. Los umbrales de sensibilidad son de 40mW/m^2 y de 100mW/m^2 para SIR y FIR respectivamente.

2.3.2.1.2. ESPECIFICACIONES MECÁNICAS.

El área que mejor cobertura ofrece para establecer una comunicación punto a punto infrarrojo, la define un sector circular con radio un radio máximo de un metro y con una apertura de ± 15º.

Las longitudes de onda seleccionadas para el estándar se encuentran ubicadas alrededor de 850 nm y 900 nm.

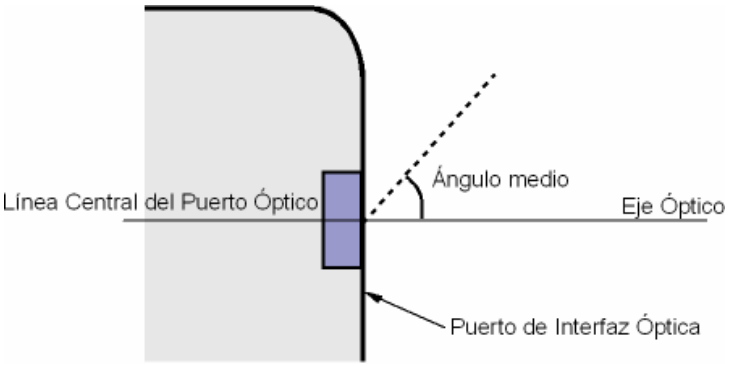


Figura 2.3.10 Geometría óptica del puerto infrarrojo. [16]

2.3.3. USB.

El Bus Serie Universal, USB, es una especificación de las empresas Compaq, IBM, Intel, Microsoft, NEC y Northern Telecom; que describen un canal serie que tolera una gran variedad de periféricos de alta, media y baja velocidad, con soporte integral para transferencias en tiempo real (isócronas) como voz, audio y video comprimido, y que permite mezclar dispositivos y aplicaciones isócronas y asíncronas.

La arquitectura USB combina todas las ventajas de un estándar multiplataforma, incluyendo un costo inferior, una mayor compatibilidad y un número superior de periféricos disponibles.

La implementación de USB elimina el uso de IRQ's[26] y canales DMA[27], así como la necesidad de apagar y abrir los equipos para instalar o quitar dispositivos, y su posterior configuración de hardware y software.

La versión USB 1.1 establece las siguientes características:

- Un acceso al bus gestionado directamente por el controlador USB, para permitir transferencias isócronas y eliminar tiempos de arbitración.

- La coexistencia de dispositivos isócronos y asíncronos. Los dispositivos isócronos se atienden en función del ancho de banda y latencia requeridos, y los asíncronos se atienden durante el tiempo restante no consumido por los dispositivos isócronos.

- Una velocidad de 12 Mbps (Full Speed) y un subcanal de 1.5 Mbps (Low Speed) para los dispositivos mas lentos. La coexistencia en un mismo sistema de dispositivos Full Speed y Low Speed se maneja mediante conmutación automática y dinámica de velocidad entre unas transferencias y otras.
- Una arquitectura fácilmente escalable para permitir la existencia de varios controladores USB en un sistema.

- Una conectividad excepcional, ya que puede manejar hasta ciento veintisiete dispositivos simultáneamente que se pueden conectar y desconectar sin tener que reiniciar el sistema (Plug & Play).

[26] IRQ: Interruption Request.
[27] DMA: Direct Memory Access.

- Una configuración automática de dispositivos, que elimina la necesidad de realizar configuraciones manuales por medio de puentes o conmutadores.

- Una distribución de alimentación desde el controlador USB, que permite la conexión tanto de dispositivos alimentados desde el bus como aquellos alimentados por fuentes externas.

La aparición de USB 2.0 responde a la necesidad continua de comunicar dispositivos que demandan mayor ancho de banda, a tal grado que esta nueva versión también llamada High Speed proporciona una velocidad de hasta 480 Mbps, cuarenta veces más rápido que las conexiones de USB 1.1.

USB 2.0 además del aumento de velocidad, establece como requisito previo la total compatibilidad con dispositivos USB 1.1, por lo que utiliza los mismos cables y conectores; y define nuevas técnicas para comunicarse con dispositivos de Low, Full y High Speed.

2.3.3.1. ESPECIFICACIONES ELÉCTRICAS.

USB utiliza una cable de cuatro conductores, de los cuales dos de ellos son para la alimentación de los dispositivos, y los otros dos transmiten/reciben una señal de datos, en modo de tensión diferencial. Los conductores de alimentación se etiquetan como V_{BUS} y GND. Entregan una tensión continua de +5V y 500mA máximo. En función de las necesidades de alimentación eléctrica de los dispositivos, estos pueden tomar la alimentación de estas líneas, o bien tener una fuente de alimentación alternativa. Los conductores de transmisión/recepción de datos se etiquetan como D+ y D-.

La comunicación es bidireccional, semidúplex y utiliza una codificación autoreloj NRZI. En éste tipo de codificación es fácil que el transmisor y el receptor se desincronicen tras el envío de varios unos sucesivos, para solucionar esto se

emplea un bit llamado Stuffing Bit, que consiste en la inserción de un cero tras la transmisión de seis estados lógicos uno, para asegurar transiciones en la línea y permitir que el receptor se mantenga sincronizado.

Los dispositivos cuentan con transmisores/receptores diferenciales y resistencias de terminación con los que pueden transmitir y detectar varios estados eléctricos distintos en la línea, definidos como:

- Transmisión/Recepción diferencial de bits: Estados DIFF0 y DIFF1, denominados también como estados J y K.

- SE0 (Single-Ended 0): Ambas señales D+ y D- a 0V. Se utiliza para detectar la conexión/desconexión de dispositivos, para indicar el EOP (Fin de Paquete) y para generar reset.

- IDLE: Reposo o línea en alta impedancia, necesario para permitir transferencias semidúplex, detectar la conexión y desconexión de dispositivos y discriminar entre dispositivos Full Speed y Low Speed.

- SOP (Inicio de Paquete): Se indica mediante una transición IDLE a K.

- EOP (Fin de Paquete): Se indica mediante una secuencia SE0 (2 bits) + J (1 bit) + IDLE.

- Detección de dispositivo y discriminación Full Speed/Low Speed: Cuando el transmisor deja la línea en IDLE, si hay un dispositivo conectado su polarización fuerza un estado J (DIFF0) si es Low Speed, por otro lado, si se conecta un dispositivo Full Speed, se fuerza un estado K (DIFF1), y si no lo hay, la polarización del transmisor fuerza un estado SE0.

- Reset: transmisión de SE0 durante ≥ 10 ms.

La tabla que se presenta a continuación detalla los niveles de las señales J y K para cada velocidad soportada en el bus.

Estados de Transmisión/Recepción.	Velocidad	Estado Lógico / Magnitud
J	Low	DIFF0: (D-) – (D+) > 200mV
	Full	DIFF1: (D+) – (D-) > 200mV
	High	DIFF1: (D+) – (D-) > 400mV
K	Low	DIFF1: (D+) – (D-) > 200mV
	Full	DIFF0: (D-) – (D+) > 200mV
	High	DIFF0: (D-) – (D+) > 400mV

Tabla 2.3.2. Niveles de voltaje para los estados de Transmisión/Recepción [18].

En modo High Speed, cada hilo de tensión diferencial esta conectado en sus extremos a una resistencia pull-down de 45 Ω. Esta resistencia es necesaria para mantener la corriente de línea en 17.78mA, ya que la amplitud de los voltajes de estados diferenciales (DIFF0 y DIFF1) de alta velocidad se conmuta a 400mV.

Los dispositivos USB 1.1 identifican su velocidad eléctricamente, mediante una resistencia de pull-up en la línea D+ (Full Speed) o D- (Low Speed). En cambio los dispositivos High Speed (USB 2.0) se identifican en principio como Full Speed y durante el proceso de reset ejecutan un protocolo de bajo nivel (determinadas secuencias de señales eléctricas de niveles especiales) a través del cuál determinan si están conectados a un puerto Full Speed o High Speed. Si el puerto es High Speed, detectará y responderá al protocolo iniciado por el dispositivo. Sólo si el concentrador y el dispositivo ejecutan el protocolo, se establece una comunicación High Speed entre ellos.

2.3.3.2. ESPECIFICACIONES MECÁNICAS.

2.3.3.2.1. CONECTORES.

USB define los tipos de conectores A y B, utilizados en el host USB y en los dispositivos, respectivamente. La diferencia radica en la forma misma del conector y la distribución de los contactos.

Existen algunas variantes a lo anteriormente descrito, por ejemplo, en algunos dispositivos un extremo del cable está soldado internamente y solo cuentan con un conector tipo A al otro extremo para conectarlo en el host. Otro ejemplo son algunos conectores propietarios utilizados en dispositivos móviles como PDA's y teléfonos celulares.

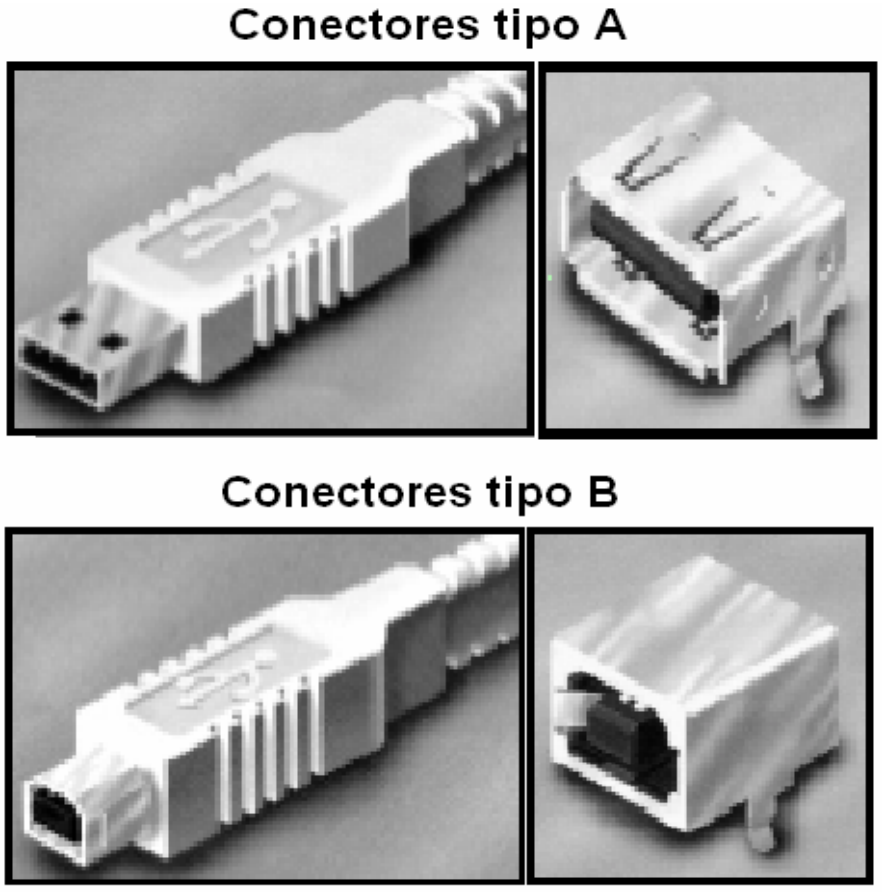


Figura 2.3.11 Tipos de conectores USB [17]

Figura 2.3.12 Ejemplo de conectores propietarios.

2.3.3.2.2. CABLE.

Tal como se menciono anteriormente, el cable USB requiere de cuatro conductores, dos orientados a la alimentación y los restantes destinados a la transferencia de la información.

Las secciones de los cables de alimentación varía entre 20 y 26 AWG, y la de los de señal diferencial es de 28 AWG. Los cables de señal diferencial siempre estarán trenzados, no así los de alimentación. Obligatoriamente para todos los tipos de cable USB, deberá contar con un apantallamiento metálico de aluminio-poliester y también con un cable trenzado de cobre de sección 28AWG, para que dote de rigidez al cable final. Además los cables para Full y High Speed contarán con un segundo apantallamiento, que consistirá en una malla de cobre estañada alrededor de todo el cable. Finalmente, y como última capa, se encuentra el recubrimiento aislante de PVC.

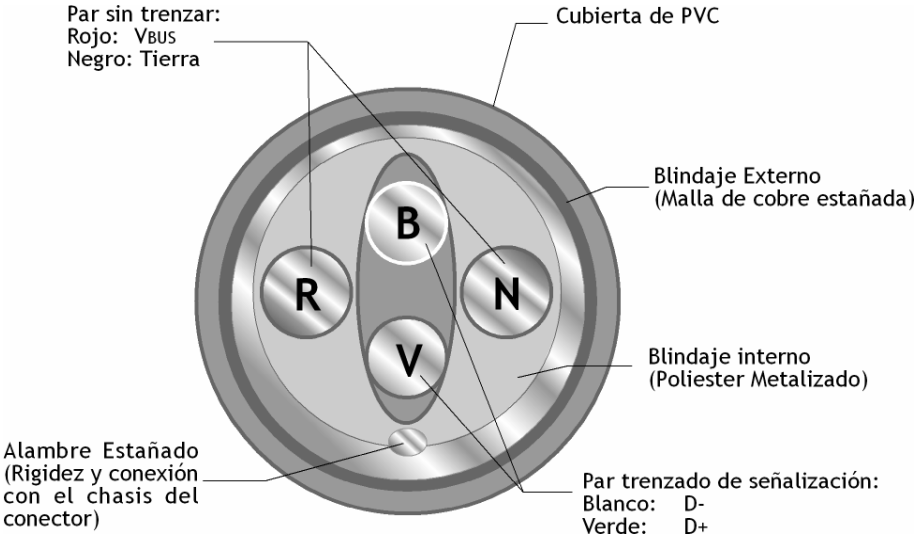


Figura 2.3.13 Sección de un cable USB Full/High Speed. [18]

También se define una longitud máxima de tres metros para dispositivos Low Speed, y una longitud de cinco metros para los dispositivos Full y High Speed. Se podrá hacer uso de alargadores, pero para ello es necesario ubicar un repetidor en cada tramo.

2.3.3.2.3. DEFINICION DE TERMINALES.

La codificación numérica y de color utilizada en cables y conectores, se presenta en la tabla 2.3.3 y la figura 2.3.14.

Número de contacto	Nombre de la señal	Asignación física de los cables
1	V_{BUS}	Rojo
2	D-	Blanco
3	D+	Verde
4	GND	Negro
Chasis del conector	Blindaje	

Tabla 2.3.3 Asignación de conectores USB. [18]

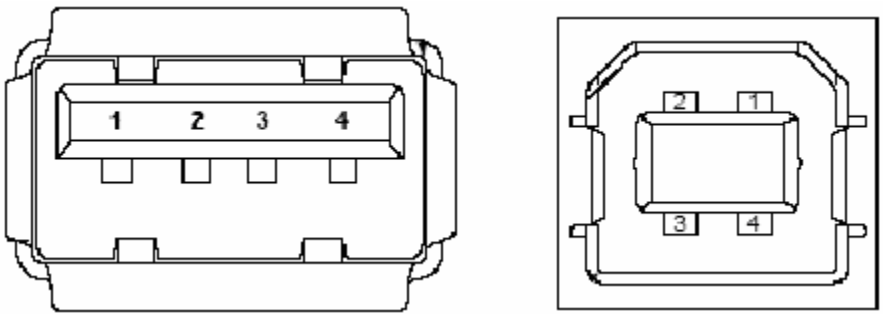


Figura 2.3.14 Identificación de los pines en conectores tipo A y B. [17]

2.3.3.3. TIPOS DE DISPOSITIVOS.

USB define tres tipos de dispositivos: Controlador, Concentradores o Hubs y periféricos; teniendo como característica común todos los dispositivos USB, la integración de los siguientes bloques funcionales:

- Transceiver[28]: Encargado de seleccionar la velocidad de comunicación del dispositivo.

- Function Interface Unit (FUI): Administración de datos basado en estados de colas FIFO[29] y envío de interrupciones.

- FIFO's: El controlador tiene ocho buffers FIFO, la mitad de ellos usados para transmitir y el resto para la recepción.

- Serial Interface Engine (SIE): Realiza la conversión de datos a formato serie, realiza la codificación digital de los bits, controla la detección de errores, maneja el protocolo de comunicación y la secuencia de paquetes.

[28] Transceiver: Transmisor/Receptor.
[29] FIFO: First Input First Output.

2.3.3.3.1. CONTROLADOR.

Reside dentro del PC y es responsable de las comunicaciones y el control del flujo de datos entre los periféricos USB y el CPU del host. Es también responsable de la admisión de periféricos dentro del bus, tanto si se detecta una conexión como una desconexión.

Para cada periférico añadido, el controlador determina su tipo y le asigna una dirección lógica para utilizarla siempre que se comunique con éste. Si se detecta un error en la conexión, el controlador lo informa al CPU, que, a su vez, lo transmite al usuario. Si no existiese problema alguno en la conexión, el controlador asigna al periférico los recursos del sistema que éste precise para su funcionamiento.

2.3.3.3.2. CONCENTRADORES O HUBS.

Son distribuidores inteligentes de datos y alimentación, y hacen posible la conexión a único puerto USB de ciento veinte y siete dispositivos. De una forma selectiva reparten datos y alimentación hacia sus puertas descendentes y permiten la comunicación hacia sus puertas de retorno o ascendentes. Los concentradores también permiten las comunicaciones desde el periférico hacia el host, aceptando datos en las puertas descendentes y enviándolas hacia el PC por la puerta de retorno.

Además del controlador, el PC también contiene el concentrador raíz. Este es el primer concentrador de toda la cadena que permite a los datos y a la energía pasar a los conectores USB presentes en el host, y de éstos a los ciento veinte y siete periféricos que como máximo puede soportar el sistema. Esto es posible mediante la adición de concentradores externos a la PC.

2.3.3.3.3. PERIFERICOS.

USB soporta tres tipos de velocidades para la comunicación, de acuerdo a las necesidades del ancho de banda requerido por los periféricos, el bus le asigna los recursos necesarios para conseguir mayores eficiencias en su rendimiento.

Los periféricos de baja velocidad como teclados y dispositivos de señalización, no requieren de 12 Mbps o 480 Mbps y empleando un ancho de banda Low Speed, se puede dedicar mas recursos del sistema a periféricos tales como dispositivos de almacenamiento masivo o extraíble, módems, scanner, cámaras digitales, reproductores de audio, entre otros, requieren velocidades mas altas para transmitir mayor volumen de datos o datos cuya dependencia temporal es mas critica.

2.3.3.4. MODELO DEL FLUJO DE DATOS EN USB.

USB define una arquitectura de tres niveles o capas (similar a una definición OSI), en ella se especifica la dirección del flujo real de datos, es decir para empaquetar y desempaquetar la información útil desde el software cliente hasta una función o viceversa y también muestra los canales lógicos de comunicación entre bloques del mismo nivel.

La figura 2.3.15 presenta una visión general del flujo de datos en el bus, en ella se destacan las diferentes capas del sistema y en particular los siguientes bloques funcionales:

- Software Cliente: Software ejecutado en el host y contraparte del dispositivo USB. Este software cliente es típicamente proporcionado por el sistema operativo o proveído con el dispositivo USB.

- Software del sistema USB: Software de soporte para USB en un sistema operativo en particular. Este software normalmente esta contenido en el sistema operativo y es independiente de un dispositivo USB en particular o software cliente.

- Controlador del host USB: Hardware y software necesarios que permiten a los dispositivos USB la conectividad con el host.

- Dispositivo físico USB: Artículo electrónico conectado al final de un cable USB y diseñado para cumplir cierta función específica.

- Interfaz del bus USB: Proporciona conectividad entre el host y el dispositivo, desde el punto de vista físico, de señalización y empaquetado de la información.

- Dispositivo Lógico USB: Es el encargado de soportar el flujo de datos ascendente y descendente para los niveles de de función e interfaz del bus USB.

- Función: Proporciona capacidad adicional para que el host encuentre de forma certera su correspondiente par al software cliente.

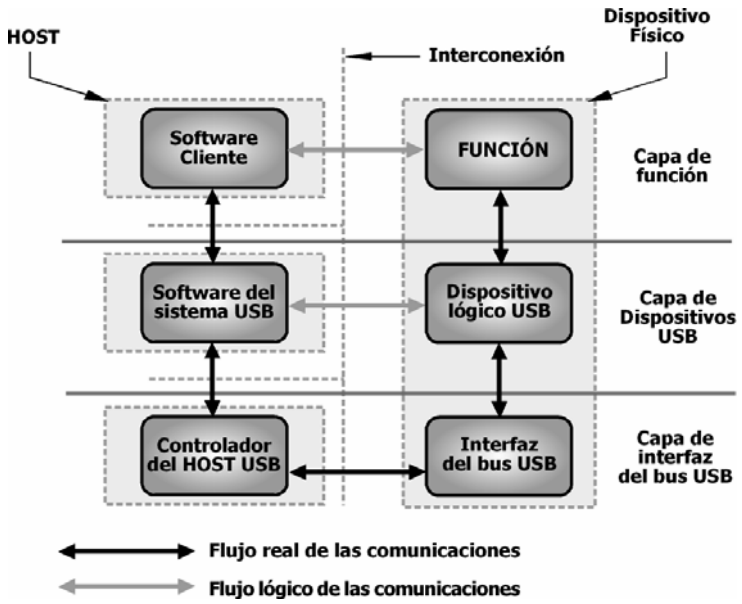


Figura 2.3.15 Esquema general del flujo de datos en bus. [18]

2.3.3.4.1. TOPOLOGÍA DEL BUS.

Para la definición de la topología USB, se considera al Host y a los Dispositivos, como los elementos primarios del sistema que interactúan enmarcados en un tipo específico de topología física, lógica.

El host USB coordina completamente el sistema, aceptando o rechazando la conexión de dispositivos, mediante el control total del acceso al bus. El host también es responsable de monitorear la topología del bus.

Los dispositivos físicos USB proveen de funciones adicionales al host, los tipos de funciones pueden ser variadas y orientadas a distintas necesidades. Sin embargo, todos los dispositivos lógicos USB (Capa de dispositivos USB) presentan la misma interfase básica de cara al host, permitiendo que el host maneje los aspectos relevantes del bus de diferentes dispositivos bajo la misma forma.

Para asistir al host en la tarea de identificar y configurar los dispositivos USB, cada dispositivo trasporta su configuración y la reporta cada vez que es aceptado en un host.

2.3.3.4.1.1. TOPOLOGÍA FÍSICA.

Los dispositivos están conectados al host, utilizando una topología de estrella escalonada, ilustrada en la figura 2.3.16. El punto central es el Hub Raíz, a partir de este se derivan puntos de conexión directos o puntos extendidos de conexión los cuales son proveídos por hubs externos.

Varias funciones pueden ser empaquetadas juntas en un mismo dispositivo físico. Dentro del paquete, cada función individual esta

conectada a un hub interno y éste se encarga de los servicios USB. Generalmente estos dispositivos se conocen como dispositivos compuestos.

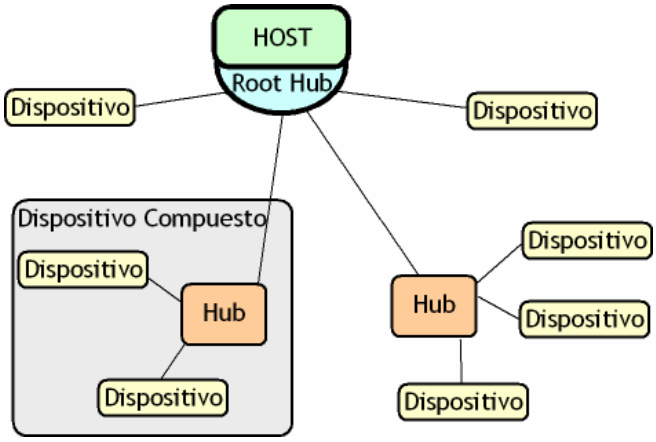


Figura 2.3.16 Topología física del bus. [18]

2.3.3.4.1.2. TOPOLOGÍA LÓGICA.

Mientras un dispositivo está conectado, atravesando todos los escalones físicos necesarios para alcanzar el bus, desde el punto de vista lógico el host se comunica con cada dispositivo como si estuvieran directamente conectados al puerto raíz, formando una configuración en estrella. Esta forma de conexión se presenta en la figura 2.3.17 y corresponde, de forma lógica, a la topología física mostrada anteriormente.

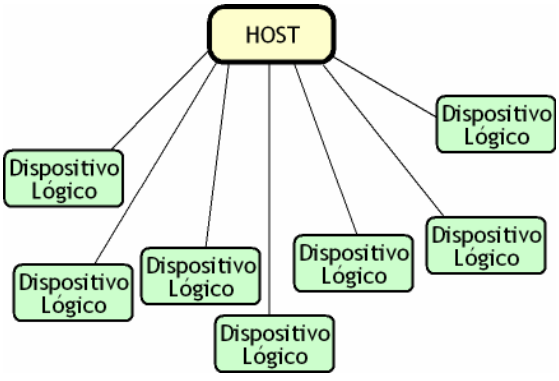


Figura 2.3.17 Topología lógica del bus. [18]

2.3.3.4.1.3. RELACIÓN SOFTWARE CLIENTE – FUNCIÓN.

Durante la operación, el software cliente debería ser independiente de otros dispositivos que estén compartiendo el bus. Por lo que cada software cliente mantiene una relación punto a punto con cada función, dando paso al establecimiento de n relaciones, como n dispositivos estén conectados al bus.

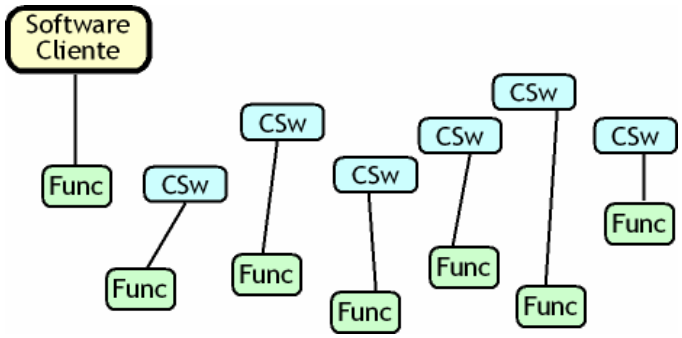


Figura 2.3.18 Relación entre el software cliente y una función. [18]

2.3.3.4.2. FLUJO DE COMUNICACIÓN USB.

El bus proporciona un servicio de comunicación entre el software de un host y una función USB. La forma real de comunicación es en dirección vertical iniciando desde la capa mas alta hacia las capas inferiores cuando se transmite y luego en sentido contrario en el equipo receptor, es decir efectuando la operación de empaquetado y luego desempaquetando la información.

El flujo de comunicación lógica, se explica mediante el uso de buffers, tuberías (pipes) y endpoints. Este modelo de comunicación es de dirección horizontal, conectando "lógicamente" entidades de la misma capa.

En la figura 2.3.19 se presenta con más detalle lo mostrado en la figura 2.3.15. En ella se puede observar el flujo real de comunicación junto con los

nombres que van tomando los datos a medida recorren cada una de las capas definidas por USB (empaquetado y desempaquetado de la información). También se ilustran los caminos lógicos, llamados tuberías, que son utilizados para establecer la comunicación entre un endpoint del dispositivo y un buffer del software ubicado en el host.

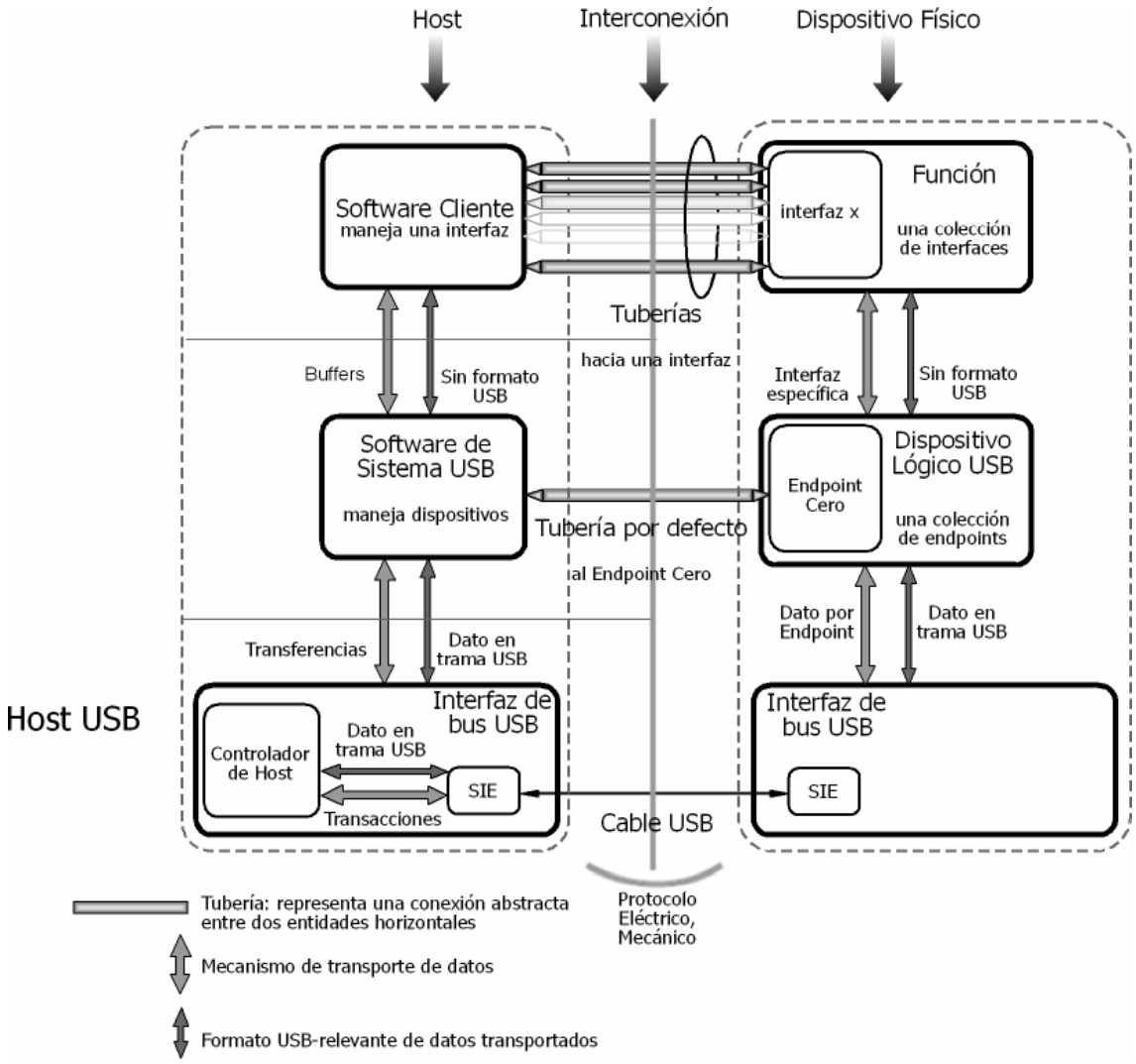


Figura 2.3.19 Flujo de datos detallado en el bus. [18]

Las entidades de la capa de dispositivos lógicos hacen uso únicamente de la tubería por defecto (Default Pipe) para establecer su comunicación entre un buffer del software del sistema USB y un grupo de endpoint en particular,

llamados Endpoint Cero. Esta tubería es utilizada para que el software del sistema USB maneje los dispositivos.

El software cliente maneja una interfaz usando tuberías (asociadas a un conjunto de endpoints). El software cliente realiza la petición para mover los datos entre un buffer en el host y un endpoint en el dispositivo, a través del bus (físicamente). El controlador del host (o el dispositivo USB, depende de la dirección de la transferencia) empaqueta los datos para ser movidos. El controlador del host también coordina cuando el bus es usado para mover paquetes de datos.

2.3.3.4.2.1. ENDPOINTS.

Un endpoint es por si solo una conexión simple, que soporta un flujo de datos de entrada o de salida. Cada dispositivo USB está compuesto por unos endpoints independientes, y una dirección única asignada dinámicamente por el sistema, una vez que el dispositivo este conectado.

A su vez, cada endpoint dispone de un identificador único dentro del dispositivo (No. de endpoint) que viene asignado de fábrica, además de una determinada orientación de flujos de datos.

Un grupo de endpoints dan lugar a la existencia de interfases, las cuales permiten controlar las funciones del dispositivo.

2.3.3.4.2.2. PIPES.

Una tubería USB es una asociación entre uno o dos endpoints en un dispositivo, y el software en el host. Las tuberías permiten mover datos

entre software en host, a través de un buffer, y un endpoint en un dispositivo. Hay dos tipos de tuberías:

- Stream: Los datos se mueven a través de la tubería sin una estructura definida, estas tuberías siempre son unidireccionales y los datos se transmiten de forma secuencial: first in, first out. Están pensadas para interactuar con un único cliente. Un stream siempre se asocia a un único endpoint en una determinada orientación.

- Mensaje: Los datos se mueven a través de la tubería utilizando una estructura USB definida. Las tuberías de mensajes permiten la comunicación en ambos sentidos. El software USB del sistema se encarga de que múltiples peticiones no se envíen a la tubería de mensajes concurrentemente. Un dispositivo ha de dar únicamente servicio a una petición de mensaje en cada instante por cada tubería de mensaje.

La tubería que esta formada por dos endpoints con número cero se denomina tubería de control por defecto. Esta tubería esta siempre disponible una vez se ha conectado el dispositivo y ha recibido un reseteo del bus. La tubería de control por defecto es utilizada por el software USB del sistema para obtener la identificación y para configurar al dispositivo.

2.3.3.4.3. TIPOS DE TRANSFERENCIA DE DATOS.

Para mover la información desde el origen hasta el destino, USB determina una estructura de datos llamada trama o micro trama (para USB 2.0). Estas tramas poseen un campo de carga para transportar los datos. Los datos aceptados en las tramas pueden tener distintos arreglos, que dentro de las especificaciones del bus, se les conoce como transferencias.

USB determina cuatro tipos de transferencias, transferencias Isócronas, de Control, de Interrupción y transferencias Bulk. Cada una de ellas considera aspectos tales como el tamaño del paquete, latencia de los datos, manejo de errores, secuencia de los datos, dirección del flujo de comunicación, etc., como parámetros para enmarcar las características de cada tipo de transferencia.

2.3.3.4.3.1. TRANSFERENCIAS ISÓCRONAS.

Este tipo de transferencia solo es utilizable por dispositivos Full Speed y High Speed. Garantiza un acceso al bus con una latencia limitada, asegura una transmisión constante de los datos a través del puerto siempre y cuando se suministren datos, además en caso que la entrega falle debido a errores no se intenta reenviar los datos.

La información útil por paquete puede oscilar entre 1 y 1,023 bytes. En cada trama se transfiere un paquete por cada conexión isócrona establecida. El sistema puede asignar como máximo el 90% del tiempo de trama (80% del tiempo de una microtrama para High Speed) para transferencias isócronas y de interrupción. En función de la cantidad de datos que se estén transmitiendo en un momento determinado, en velocidad media el porcentaje de transmisión utilizado puede variar desde un 1% hasta un 69%, mientras que el porcentaje de velocidad alta varía entre un 1% y un 41%.

2.3.3.4.3.2. TRANSFERENCIAS DE CONTROL.

Las transferencias de Control proporcionan control de flujo y una entrega de datos garantizada y libre de errores. Se procesan por medio de un mecanismo "best effort", es decir que el controlador USB procesa la transferencia en función del tiempo disponible en cada trama.

Como mínimo se reserva el 10% del tiempo de trama para este tipo de transferencia (Low Speed y Full Speed) y el 20% de una microtrama para dispositivos High Speed. Se puede utilizar tiempo adicional siempre que las necesidades de los tráficos isócrono y de interrupción lo permitan.

Las transferencias de Control se componen de tres transacciones denominadas como:

- Transacción de configuración: En esta se envía al dispositivo un paquete que especifica la operación a ejecutar. Ocupa 8 bytes para todas las velocidades.

- Transacción de datos: Se transfieren los paquetes de datos en el sentido indicado por la transacción de configuración. La información útil por paquete varia de acuerdo a la velocidad del dispositivo, se define para High Speed un tamaño de 64 bytes, mientras que para Full Speed, se puede fijar tamaños de 64, 32, 16 u 8 bytes; y únicamente 8 bytes por paquete para dispositivos Low Speed.

- Transacción de estado: El receptor informa el estado final de la operación.

2.3.3.4.3.3. TRANSFERENCIAS DE INTERRUPCIÓN.

Las transferencias de Interrupción están diseñadas para soportar aquellos dispositivos que precisan enviar o recibir datos de manera no frecuente, pero con ciertos límites de latencia. Aseguran una transacción (paquete) dentro de un periodo máximo, los dispositivos Low Speed pueden solicitar de 10 a 255 ms, los Full Speed de 1 a 255 ms y los dispositivos High Speed requieren de 125 μs a 4.096 seg. Esta transferencia garantiza el máximo

servicio para el puerto durante el periodo en el que envía. Incorpora detección de errores y retransmisión de datos.

Las transferencias de Interrupción se componen sólo de transacciones de datos. El tamaño del paquete de datos máximo es de 1,024 bytes para alta velocidad, 64 bytes para velocidad media, 8 bytes para baja velocidad y en High Speed/ High Bandwidth, 2 ó 3 transacciones por microtrama de hasta 1,024 bytes cada una.

2.3.3.4.3.4. TRANSFERENCIAS BULK.

Al igual que las transferencias isócronas, las transferencias Bulk solo se aplican a dispositivos de velocidades medias y altas. La información útil por paquete puede ser de 8, 16, 32 o 64 bytes para Full Speed; mientras que para High Speed se delimita a 512 bytes.

Esta diseñada para dispositivos que necesitan transmitir grandes cantidades de datos en un momento determinado sin importar mucho el ancho de banda disponible en ese momento. Esta transferencia garantiza el acceso al USB con el ancho de banda disponible, además en caso de error se garantiza el reenvío de los datos. Por lo tanto este tipo de transferencia garantiza la entrega de los datos pero no un determinado ancho de banda o latencia.

Este tipo de operación se conoce como “Good effort”, en el que el sistema aprovecha cualquier ancho de banda disponible y en el momento en que esté disponible.

2.3.3.5. PROTOCOLO DE COMUNICACIÓN.

El protocolo de comunicaciones del bus USB entre el host y el dispositivo físico se realiza a través de tokens (testigos). De forma general toda función en principio esta esperando a que el host le envíe un paquete, si este paquete es de tipo token entonces cambia el estado de la función iniciándose una transacción. El controlador USB, que es el encargado de la comunicación entre los periféricos USB y el procesador del host, va a ser el que transmita esos tokens con la dirección del dispositivo y demás información necesaria.

Además, USB establece una unidad de tiempo base equivalente a 1ms denominada frame y aplicable a buses de baja y media velocidad, en altas velocidades se trabaja con microframes, que equivalen a 125μs. Los frames no son más que un mecanismo del bus para controlar el acceso a este, en función del tipo de transferencia que se realice.

2.3.3.5.1. FORMA DE TRANSMISIÓN.

La forma en la que las secuencias de bits se transmiten en USB es la siguiente; primero se transmite el bit menos significativo, después el siguiente menos significativo y así hasta llegar al bit mas significativo. Cuando se transmite una secuencia de bytes se realiza de la misma manera.

En la transmisión se envían y reciben paquetes de datos, cada paquete de datos viene precedido por un campo Sync y acaba con el delimitador EOP (End of Packet), todo esto se envía codificado además de los bits de relleno insertados. Cuando se habla de datos se refiere a los paquetes sin el campo Sync ni el delimitador EOP, y sin codificación ni bits de relleno.

2.3.3.5.2. FORMATO DE LOS PAQUETES.

El primer campo de todo paquete de datos es el campo PID (Packet Identifier Field). El PID es una unidad de información que precede a cualquier trama USB, consta de 8 bits, los cuatro primeros son empleados para indicar el tipo de trama o paquete, interfaz USB, formato del paquete y los cuatro últimos para el mecanismo de detección de errores empleado sobre la trama.

2.3.3.5.2.1. TIPOS DE TRAMA.

Existen tres tipos de tramas o paquetes, el uso de cada uno de ellos está en función del tipo de comunicación.

- Token Packets: Identifican el destino/origen de los paquetes subsecuentes. Un token packet está constituido por un PID donde se especifica el tipo de token (IN, OUT, SETUP y STO), un campo ADDR, ENDP y un CRC. En transacciones OUT y SETUP los campos ADDR y ENDP identifican al destinatario que recibirá los siguientes paquetes de información. En transacciones IN estos campos identifican al dispositivo USB que debe transmitir información. Los tokens del tipo STO (Start-of-Frame) proporcionan información de sincronización, estos paquetes preceden a cada trama indicando su secuencia.

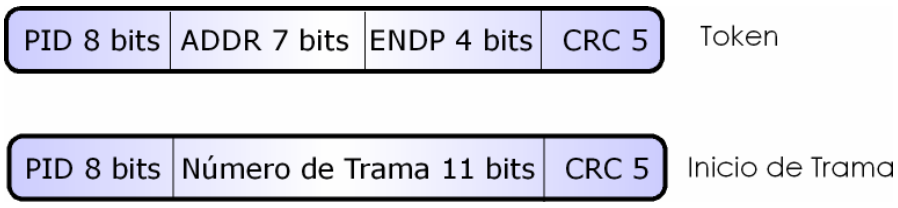


Figura 2.3.20 Tramas de tokens. [18]

- Handshake Packets: Se utilizan para la negociación y el estado de la comunicación. Estos paquetes consisten únicamente en un PID, estos se

emplean para reportar el estado de la transmisión y pueden devolver valores indicando la recepción satisfactoria de datos, comandos de aceptación o rechazo, control de flujo y situaciones parada. Se identifican tres tipos de paquetes de Handshake:

- ACK. Indica que el paquete de datos fue recibido correctamente sin errores.
- NAL. Indica que un envío de datos no ha sido aceptado.
- STALL. Indica que un dispositivo es incapaz de transmitir o recibir datos.

- Data Packets: Intercambio de información. El paquete esta constituido por un PID, un campo con los datos contenidos y un CRC. Este último es generado sobre la porción de datos.

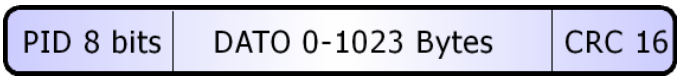


Figura 2.3.21 Trama de datos. [18]

2.4. TELÉFONOS MÓVILES GSM.

Las características constructivas y operativas de los teléfonos utilizables en los sistemas celulares difieren notablemente de los equipos terminales conectados a la red telefónica conmutada pública, los teléfonos móviles son dispositivos electrónicos con diseños compactos e intrincados, con partes encargadas de procesar cálculos para comprimir y descomprimir el flujo de voz, asegurar la transmisión y recepción de la información y controlar aplicaciones extras proveídas por el propio teléfono.

2.4.1. COMPONENTES BÁSICOS.

Típicamente todos los teléfonos móviles, cuentan con cinco componentes esenciales:

- El microteléfono, contiene todos los elementos de interacción con el usuario: transductores acústicos, de marcación y de funciones (repetición de marcación, envío, borrado, memoria, final de llamada) y visualizador. En realidad el microteléfono contiene también la unidad de control de todo el radioteléfono y comanda el resto del equipo.

- El transceptor, consiste en un radiotransmisor y receptor que utiliza un sintetizador de frecuencia para sintonizar cualquier canal de radio del sistema celular. La unidad lógica del transceptor interpreta los comandos emitidos por el microteléfono y gestiona los circuitos de radio. Se comunica también con las estaciones de base para establecer las conexiones, determinar las frecuencias adecuadas y coordinar el cambio de célula. El transceptor presenta interfaces de conexión al microteléfono, a la antena y a al sistema de energía.

- La antena es el elemento encargado de transferir la señal electromagnética portadora de la comunicación desde y hacia el teléfono, es un elemento crítico y su tipo y emplazamiento determinan la calidad de transmisión y recepción.

- La batería esta destinada a proporcionar autonomía energética al teléfono móvil.

- Tarjeta SIM (Subscriber Identity Module), La tarjeta SIM desempeña dos funciones primarias en la red GSM: la de control de acceso a la red (autenticación y cifrado) y la personalización del servicio (SMS, indicación de gasto de llamada, etc.). Contiene la información sobre el abonado, la seguridad y la memoria para una lista de teléfonos.

En las figuras siguientes se muestran las partes de un teléfono móvil GSM de generación 2.5, es de notar que el microteléfono y el transceptor se encuentran ubicados en un mismo circuito.

Figura 2.4.1 Partes de un teléfono móvil GSM/GPRS.

2.4.2. CARACTERÍSTICAS TÍPICAS.

Actualmente los sistemas de telefonía móvil basados en GSM, se encuentran en etapa de transición (evolución de generación 2.0 a 2.5), por lo que es común encontrar teléfonos con prestaciones orientadas hacia ambas redes, entre ellas, las más representativas son:

- Tipo de Sistema Operativo (Nokia OS, Symbian, Microsoft Mobile).

- Soporte para ejecutar aplicaciones basadas en Java2ME™.
- Cantidad de memoria disponible para el usuario (ejecución de aplicaciones, almacenamiento de contactos y contenido multimedia).
- Soporte para la conexión a redes de datos (GPRS y/o HSCSD).
- Navegador WAP integrado.
- Sistema de mensajería (SMS y/o MMS).
- Soporte para la reproducción de formatos multimedia.
- Microcámara digital.
- Resolución y profundidad de colores del visualizador.
- Funciones especiales y distribución del teclado.
- Conectividad local (Serie, USB, IrDA, Bluetooth).
- Bandas de frecuencias operativas (900 MHz, 1800 MHz, 1900 MHz)

2.5. REDES GSM.

2.5.1. SISTEMA DE TELEFONÍA CELULAR.

Aunque la idea básica de la radio celular se originó en los laboratorios Bell en 1947, no se puso en práctica hasta comienzos de 1980, por razones tecnológicas principalmente. La principal ventaja de un sistema celular sobre otros sistemas móviles de radio es su capacidad para manejar mayores cargas de tráfico debido a la reutilización eficiente de las frecuencias disponibles en el espectro radioeléctrico.

2.5.1.1. ESTRUCTURA DE CELULA.

El área a cubrir se divide en un número de áreas pequeñas, llamadas celdas o células. Cada célula se equipa con su propia estación radio base. Las células están agrupadas en clusters y el número de canales radio disponibles es distribuido en el grupo de células de manera que esta distribución se repite en toda la zona de cobertura. Esta técnica permite la reutilización de los radiocanales.

El número de células en un cluster tiene que ser determinado de manera que pueda repetirse de forma no interrumpida en el área de cobertura. Solamente algunas configuraciones lo permiten. Las agrupaciones típicas se basan en 4, 7, 12 o 21 células. Las figuras 2.5.1 y 2.5.2, presentan clusters comunes y un esquema del patrón de reutilización de frecuencias en un área con clusters de 7 celdas, respectivamente.

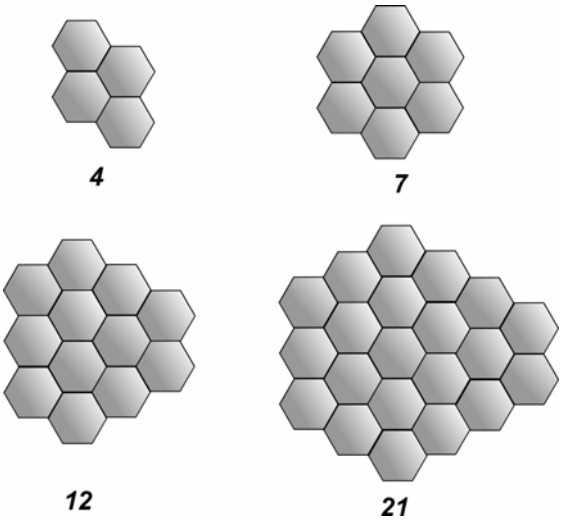


Figura 2.5.1 Patrones de repetición en un sistema de radiotelefonía móvil. [19]

El número de células en cada cluster tiene significativa importancia en la capacidad total del sistema. Cuanto menor es el número de las células, mayor es el número de canales por célula y, en consecuencia, el tráfico es más alto. Sin embargo, debe buscarse el punto de equilibrio. Si se utilizan más canales por célula y el tamaño del cluster es menor, la distancia entre las células que utilizan los mismos canales es menor, con la consecuencia de que la interferencia entre agrupaciones adyacentes aumenta (interferencia cocanal).

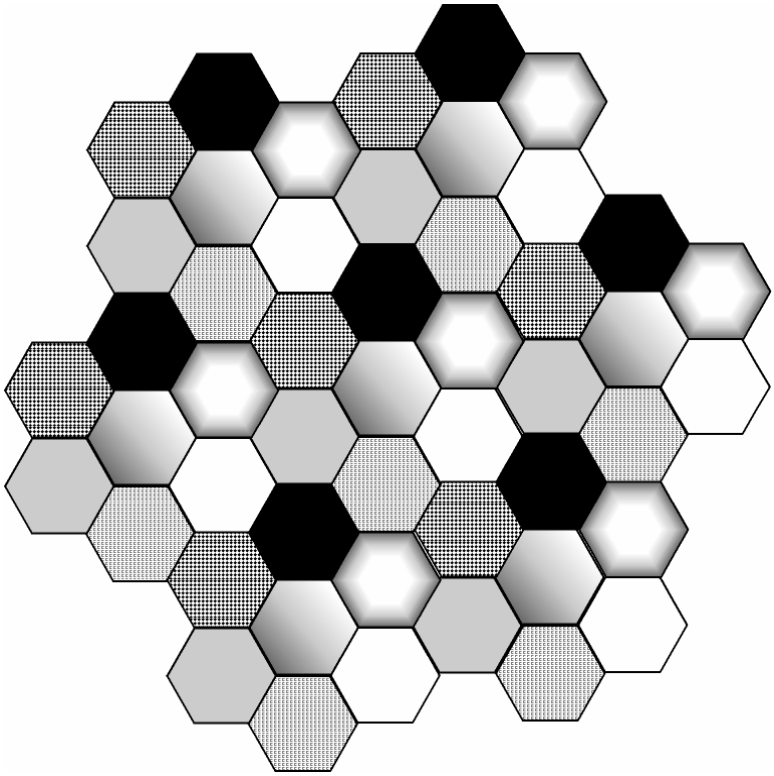


Figura 2.5.2 Reutilización de frecuencias en un cluster de 7 celdas. [20]

El número total de canales por célula, y en consecuencia el tráfico, depende del número de canales disponibles y el tipo de cluster, es decir,

$$\text{Número de canales por célula} = \frac{\text{Número total de canales}}{\text{Tamaño del cluster}} \qquad \text{Ecuación 2.7}$$

Sin embargo, el tráfico de un área en particular puede ser aumentado, cuidando los problemas de interferencia, si se reduce el tamaño de la célula, de manera que aumenta el número total de radiocanales disponibles en el área.

2.5.1.2. ESTRUCTURA DEL SISTEMA.

Las estaciones de base ubicadas en el centro de cada célula están unidas a una central de conmutación, que en esencia es una central telefónica de conmutación modificada para el sistema celular. Una red celular consistirá, en la práctica, en

varias centrales de conmutación interconectadas entre sí, ver figura 2.5.3. Esta configuración permite la realización completa de los distintos tipos de llamada, como son llamadas de móvil a fijo, de fijo a móvil y de móvil a móvil.

El sistema celular incluye dos prestaciones importantes a la hora de permitir el mantenimiento de las comunicaciones con el móvil. La primera de ellas se denomina registro y se trata de la facultad del sistema para conocer la ubicación del móvil en todo momento dentro de la zona de cobertura. La segunda característica es la capacidad del sistema para realizar el cambio de célula sin pérdida de la comunicación.

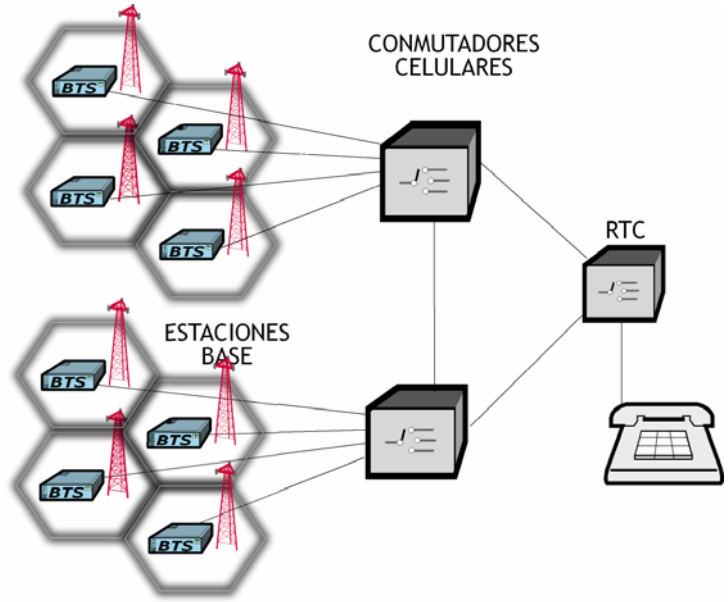


Figura 2.5.3 Estructura de un sistema de radiotelefonía celular. [19]

Dentro del sistema, se reservan un número de canales radio como canales de señalización. Adicionalmente la red se divide en un número de áreas de tráfico, cada área consiste en un grupo de células. La estación de base genera un código de identificación correspondiente con el área de tráfico a la que pertenece, como parte de la información transmitida por los canales de señalización.

El abonado móvil que viaja a lo largo de la red, monitoriza el canal de señalización cuya emisión es más potente. Como el móvil se desplaza de una célula

a la siguiente, detecta un deterioro en la calidad de recepción en el canal común de señalización utilizado y, en consecuencia, comienza la búsqueda de otro canal con señal más potente.

Una vez que el móvil ha sintonizado la nueva señal, hay dos opciones posibles. La primera es que habiendo cambiado de célula no se abandone el área de tráfico. En este caso, en lo que respecta al registro no se toma ninguna iniciativa. La segunda opción es que el móvil no haya cambiado únicamente de célula, sino también de área. En este caso el móvil transmite su identificación a la nueva estación de base, que transfiere la información al centro de conmutación. De esta manera el móvil ha registrado su ubicación de manera que la red es capaz de encaminar la llamada hacia el móvil de modo rápido y eficiente.

La segunda característica del sistema celular se denomina en inglés in call hand off. A medida que el móvil se mueve en el área de cobertura, puede cruzar la frontera entre células mientras la llamada está en progreso. Puesto que la conversación no se interrumpe hasta que se cambia de célula, la estación de base supervisa la señal recibida del móvil y detectará cualquier deterioro de la señal en los bordes de la célula. En este momento la estación de base informará al móvil que es necesario el cambio de célula (hand off). La central de conmutación ordena a las estaciones de base de las células adyacentes que supervisen la señal del móvil y elige la mejor célula a la que transferir la llamada. En la nueva célula se busca un canal de conversación libre y se instruye al móvil, todavía en la célula original, para que seleccione el nuevo canal. La parte final de la conmutación de célula dura típicamente menos de medio segundo y el usuario no percibe apenas nada extraño.

Esta pequeña interrupción de la voz no afecta a la conversación, pero sí a la transmisión de datos. En consecuencia, los dos equipos de transmisión de datos deben incluir protocolos de detección y corrección de errores para garantizar la integridad de la información.

2.5.2. GSM.

En los comienzos de los años ochenta, muchos países en Europa habían desarrollado su propio sistema de telefonía celular análoga que impedía la interoperabilidad más allá de las fronteras de cada país. En 1982, el CEPT (Conference of European Post and Telecommunication) estableció un grupo de trabajo para desarrollar un sistema paneuropeo al que se denominó GSM siglas de Groupe Speciale Mobile.

El grupo propuso desarrollar un nuevo sistema inalámbrico móvil con las siguientes premisas: itinerancia (roaming) internacional, soporte para la introducción de nuevos servicios, eficiencia espectral y compatibilidad con ISDN. En 1989, la responsabilidad por el desarrollo de GSM fue transferida al ETSI (European Telecommunications Standards Institute) que denominó al proyecto como Global System for Mobile Communications.

La evolución de GSM ha estado marcada por tres fases de evolución, la fase 1, en la que se produjeron sus especificaciones; la fase 2, en la que se propuso la inclusión de servicios de datos y de fax; y finalmente, la Fase 2+, en la que se realizan mejoras sobre la codificación de voz y se implementan servicios de transmisión de datos avanzados, entre ellos GPRS y EDGE.

GSM es un sistema de conmutación de circuitos, diseñado originalmente para voz, al que posteriormente se le adicionaron algunos servicios de datos: servicio de mensajes cortos, un servicio de entrega de mensajes de texto de hasta 160 caracteres y un servicio de datos GSM, que permite una tasa de transferencia de 9.6 kbps.

2.5.2.1. ARQUITECTURA DE LA RED GSM.

La arquitectura de la red GSM está básicamente dividida en tres partes: el sistema de conmutación, el sistema de estaciones base y el sistema de operación y mantenimiento. La figura 2.5.4 presenta un diagrama de bloques del sistema GSM.

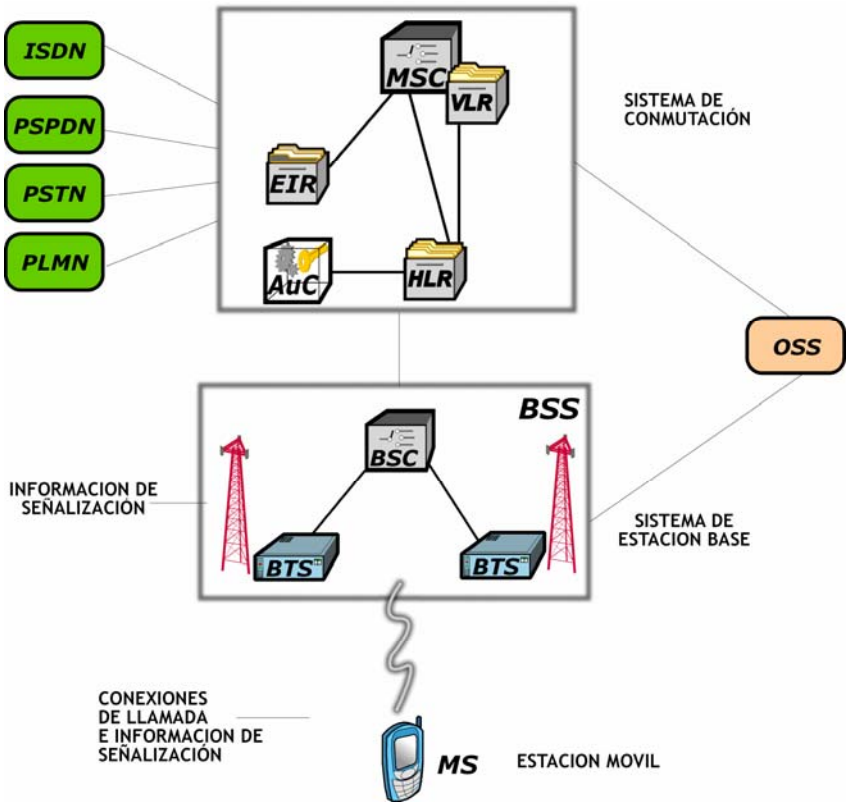


Figura 2.5.4 Modelo del sistema GSM. [19]

Cada uno de estos sistemas contiene una serie de unidades en las cuales se realizan todas las funciones que el sistema GSM es capaz de proporcionar. Las funciones relacionadas con el proceso de llamadas y abonados están implementadas en el sistema de conmutación, mientras que las funciones relacionadas con la radio se concentran en el sistema de estaciones base; todo ello está supervisado por el sistema de operación y mantenimiento. Al sistema de

estaciones base irá conectada la estación móvil vía una interfaz aérea y, a través de esta estación, el abonado de la red móvil será capaz de efectuar y recibir llamadas.

Para la gestión de llamadas hacia o desde abonados de la red fija es necesario que el sistema de conmutación tenga implementadas las interfaces apropiadas de interconexión con toda la variedad de redes fijas existentes: red telefónica básica (PSTN), red digital de servicios integrados (ISDN), red de paquetes, etc. Para la gestión de llamadas hacia o desde otros abonados móviles es necesario que el sistema de conmutación tenga implementada la interfaz hacia otras entidades de la red GSM.

2.5.2.1.1. SISTEMA DE CONMUTACIÓN.

El sistema de conmutación realiza todas las funciones normales en telefonía, como la gestión de llamadas, control de tráfico, análisis de numeración, tarificación y estadísticas de llamadas. Incluye las siguientes unidades funcionales o nodos de la red GSM:

- Central de conmutación de móviles (MSC).
- Registro de posiciones base (HLR).
- Registro de posiciones visitado (VLR).
- Centro de autentificación (AUC).
- Registro de identificación de estaciones móviles (EIR).

2.5.2.1.2. SISTEMA DE ESTACIONES BASE.

El sistema de estación base, fundamentalmente, es responsable de las funciones de radio en el sistema GSM, es decir, gestión de las comunicaciones radio, manejo del traspaso de llamadas entre células en el área bajo su control,

control del nivel de potencia de la señal tanto de las estaciones base como de las estaciones móviles etc. Incluye las siguientes unidades funcionales:

- Controlador de estaciones base (BSC).
- Estaciones base (BTS).

2.5.2.1.3. SISTEMA DE OPERACIÓN Y MANTENIMIENTO.

El sistema de operación y mantenimiento, centralizado y remoto, proporciona los medios necesarios para poder llevar a cabo una eficiente gestión de la red tanto de la parte de conmutación como de la de radio.

Las principales tareas a realizar por este sistema son: gestión de la red celular, administración de abonados, gestión de averías y medidas de funcionamiento de la red de conmutación y de radio.

2.5.2.2. DESCRIPCIÓN DE LOS NODOS DE LA RED GSM.

El sistema GSM esta compuesto por siete nodos distribuidos en los sistemas de estaciones base y de conmutación, a continuación se describirá algunas de las funciones de cada nodo.

2.5.2.2.1. MOBILE SWITCH CENTER (MSC).

La central MSC es la interfaz entre la red GSM y las redes públicas de voz y datos. Las funciones más importantes que realiza son:

- Establecimiento, enrutamiento, control y terminación de las llamadas.
- Gestión de handover (traspaso de llamadas) entre centrales.

- Gestión de servicios suplementarios.
- Recolección de datos de tarificación y contabilidad.

2.5.2.2.2. HOME LOCATION REGISTER (HLR).

Este registro es una base de datos donde se almacenan parámetros de los abonados móviles. Una red GSM puede tener uno o más HLR dependiendo de la capacidad de los equipos y de la organización de la red. Entre otros se almacenan los siguientes datos:

- Información de la suscripción.
- Información para el enrutamiento de llamadas hacia la central donde el móvil está localizado.
- Número internacional de la estación móvil IMSI.
- Número de abonado MSISDN.
- Información sobre teléservicios y servicios portadores.
- Restricciones.
- Servicios suplementarios.
- Tripletas.

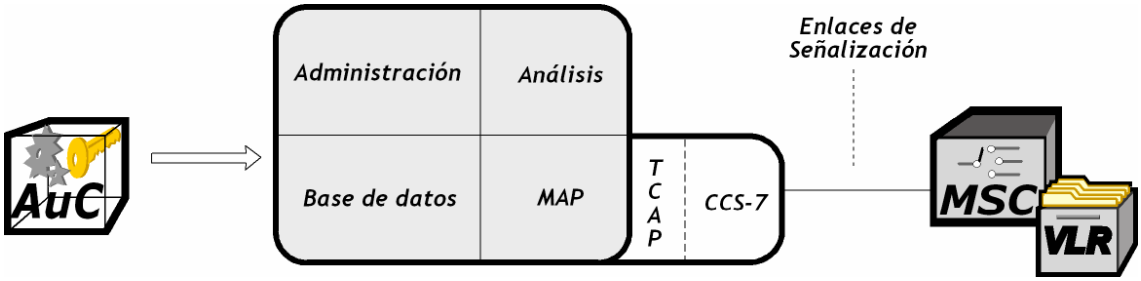


Figura 2.5.5 Estructura de HLR. [19]

En la figura 2.5.5 puede verse la estructura del HLR, que está constituido por los siguientes subnodos:

- Base de datos donde se almacenan datos y tripletas para cada abonado.

- Administración, gestiona la información hacia y desde los operadores (altas y bajas de abonados).
- Análisis, efectúa las traslaciones entre IMSI y MSISDN.
- MAP. Parte de aplicación de móviles. Es la parte de usuario que recibe y envía mensajes y toma las decisiones apropiadas. Corresponde a los niveles altos de la capa OSI (Open System Interconnection).
- CCS-7. Señalización por canal común. Permite, a través de los enlaces y terminales de señalización la transferencia de información hacia desde MSC/VLR.
- TCAP. Parte de aplicación de las capacidades transaccionales. Proporciona una interfaz estándar entre los mensajes de señalización (MAP) y la señalización propiamente dicha (CCS-7).
- Interfaz hacia el centro de autentificación para pedir tripletas.

2.5.2.2.3. VISTOR LOCATION REGISTER (VLR).

Este registro es una base de datos donde se almacenan parámetros de todos los abonados que se encuentran dentro del área de servicio del VLR.

Cuando un abonado cambia de área de servicio, el nuevo VLR debe actualizar los datos de este abonado y pide a HLR todos los datos necesarios para el establecimiento de llamadas hacia/desde el abonado móvil. Este nodo está habitualmente integrado en la MSC (MSC/VLR).

2.5.2.2.4. AUTHENTICATION CENTER (AUC).

La misión del AUC es generar tripletas para cada abonado. Las tripletas constan de:

- RAND: Número aleatorio

- SRES: Respuesta
- Kc: Clave de cifrado

La tripleta se utiliza para autentificar una llamada y para obtener las claves de cifrado de la interfaz radio. Cuando un abonado móvil intenta acceder al sistema bien porque quiere hacer o recibir una llamada o porque cambia de área de localización (cambio de VLR), se arranca de forma automática el proceso de autentificación que se indica en la figura 2.5.6.

El MS envía IMSI (Número de identificación del abonado dentro de la red móvil) hacia MSC/VLR que tiene almacenadas una serie de tripletas para cada abonado visitante. MSC/VLR envía RAND hacia MS que a su vez calcula SRES y lo envía hacia MSC/VLR que controla si coincide con el que ya tenía. Después de este control MSC/VLR decide continuar o no con la llamada.

Independientemente de este proceso, MS también calcula la clave de cifrado Kc para cifrar y descifrar la interfaz aire.

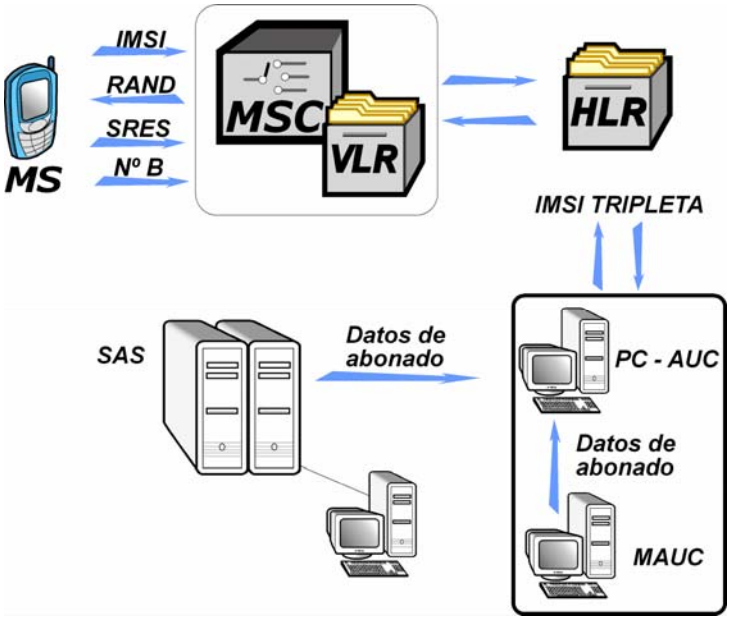


Figura 2.5.6 Proceso de autentificación. [19]

2.5.2.2.5. EQUIPMENT IDENTITY REGISTER (EIR).

Base de datos que almacenan la identidad internacional del equipo móvil (IMEI). Contiene tres listas: blanca, gris y negra, donde se clasifican datos relativos al equipo móvil.

2.5.2.2.6. BASE STATION CONTROLLER (BSC).

Hace de interfaz entre el sistema de estaciones base y el sistema de conmutación, es decir separa las funciones de radio de las de conmutación. Las principales funciones que realiza son:

- Gestión de los canales de radio.
- Supervisión de las estaciones base.
- Traspaso entre canales de la BSC.
- Gestión de la transmisión hacia las estaciones base.
- Transcodificador y adaptador de velocidades.
- Localización de las estaciones móviles.

2.5.2.2.7. BASE TRANSCEIVER STATION (BTS).

Incluye la interfaz radio y los equipos de transmisión necesarios para cubrir una o varias células. Las funciones más importantes son:

- Codificación/decodificación de los canales
- Cifrado/descifrado del camino radio.
- Medidas de la intensidad de la señal.
- Diversidad en recepción.
- Búsqueda del MS.
- Recepción de las peticiones de canal desde MS.

2.5.2.3. INTERFAZ RADIO.

La interfaz radio es el nombre con el que se conoce la conexión entre la estación móvil (MS) y la estación base (BTS). GSM utiliza una combinación de TDMA y FDMA. Dos bandas de frecuencias, de 25 MHz cada una, han sido asignadas a GSM-900, estas bandas son usadas en modo FDD, utilizando las frecuencias siguientes:

- Enlace ascendente 890 - 915 MHz (de la MS a la BTS)
- Enlace descendente 935 - 960 MHz (de la BTS a la MS)

Cada banda se encuentra dividida en canales portadores de 200 kHz de tamaño (124 portadoras).

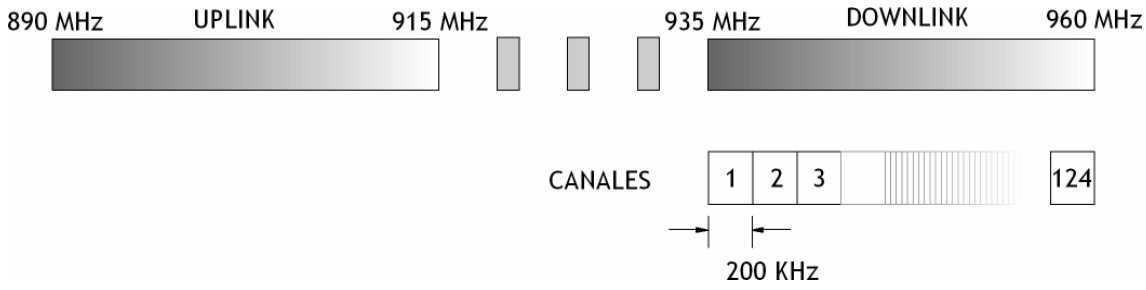


Figura 2.5.7 Enfoque FDMA utilizado en GSM. [21]

En GSM, el enfoque TDMA es aplicado a los canales ascendentes y descendentes, cada canal es dividido en ocho ranuras (slots) en cada una de las cuales se transmite una unidad de información. Este proceso se muestra en la figura 2.46.

La codificación de la señal se hace utilizando CODEC vocales que permiten transmisión de voz a 13 kbps (22,8 kbps porque necesita redundancia para detección de errores) y se utiliza una técnica de modulación digital denominada

GMSK[30], con esta modulación, se pueden alcanzar tasas de bits de 270 kbps aproximadamente. Los datos en una ranura son denominados ráfagas (burst) y alcanzan los 148 bits de longitud, incluyendo 3 bits de cabecera y 3 bits de cola, por lo que 8.25 bits restantes, son utilizados como guardas en el tiempo.

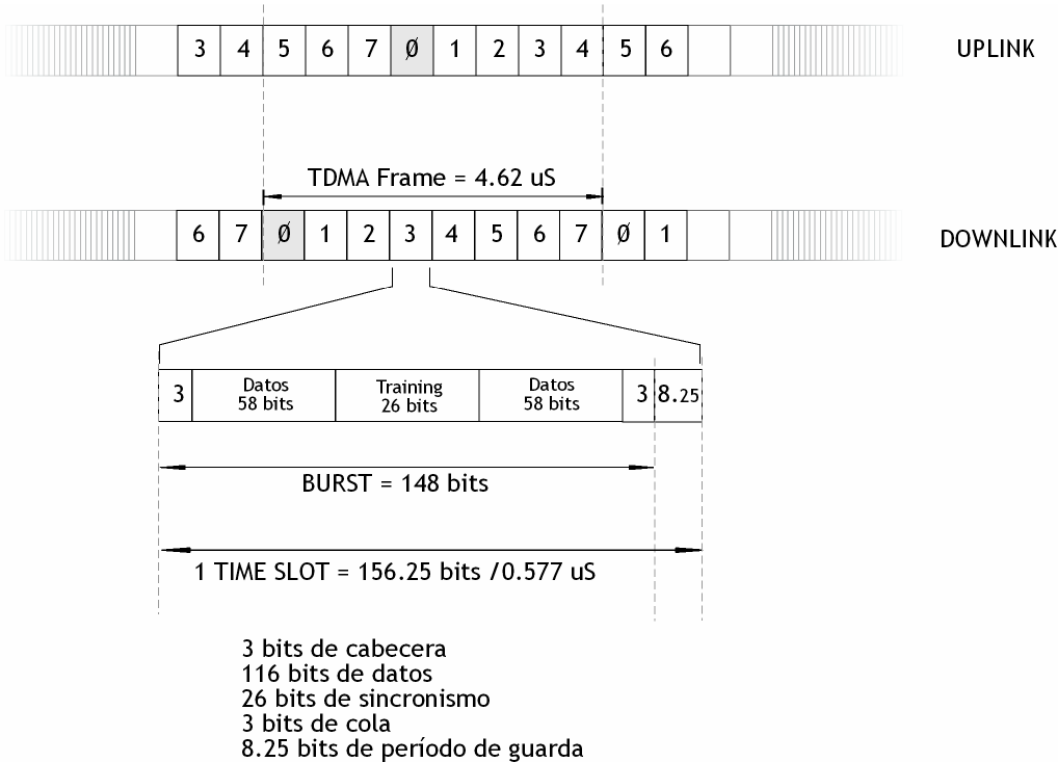


Figura 2.5.8 Tramas TDMA utilizadas en GSM. [21]

Como se puede comprobar en la figura anterior, entre los canales asignados a una misma comunicación en los enlaces ascendente y descendente existen tres intervalos de tiempo. De manera que un terminal nunca recibe y transmite al mismo tiempo con lo que se consigue una comunicación duplex sin necesitar filtros duplexores.

A través de esta interfaz se puede enviar una gran variedad de información (datos de abonado, señalización de control, etc.). Dependiendo del tipo de

[30] GMSK: Gaussian Minimum Shift Keying

información transmitida se habla de diferentes canales lógicos que se envían a través de los canales físicos.

GSM distingue entre canales físicos (las ranuras de tiempo) y canales lógicos (la información portada por los canales físicos). Algunas ranuras de tiempo en una portadora constituyen un canal físico el cual es usado por diferentes canales lógicos para transferir información, tanto de señalización como del usuario. Existen dos tipos de canales lógicos en GSM: Los canales de tráfico, TCH (Traffic Channels), que transportan información (voz o datos) del usuario y los canales de control, CCH (Control Channels), que transportan señalización y sincronización entre la estación base y la estación móvil. Sus funciones y formas varían según el enlace.

2.5.2.3.1. CANALES DE TRÁFICO.

Canales de tráfico (TCH), que contienen la información del usuario, o sea voz codificada o datos. En un principio se utilizan canales de 22,8 kbps, aunque en el futuro está prevista la utilización de canales a velocidad mitad (11,4 kbit/s) que permitirán doblar la capacidad del sistema.

2.5.2.3.2. CANALES DE CONTROL.

Canales de control, que contienen la señalización o datos de sincronización. Se dividen a su vez en:

- Canales de radiodifusión (BCH): Transmiten de la BTS hacia varias MS información sobre correcciones de frecuencia, sincronización, etc.

- Canales comunes de control (CCCH). Transmiten punto a punto (del BTS a un único MS y viceversa) información para localización del móvil, petición de acceso, etc.

- Canales de control dedicados (DCCH). Sirven para la inicialización de las llamadas, envío de informes sobre medidas de potencia o señalizaciones especiales como en el caso de un handover o traspaso entre células.

Como ya se ha mencionado, la información que transporta un intervalo de tiempo se conoce como ráfaga. Hay varios tipos de ráfagas: normal (para TCH y la mayoría de los canales de control), de corrección de frecuencia, de sincronización, de acceso y de falsa trama (esta última no contiene información).

Un ejemplo de cómo los canales lógicos se envían a través de los canales físicos aparece en la figura siguiente.

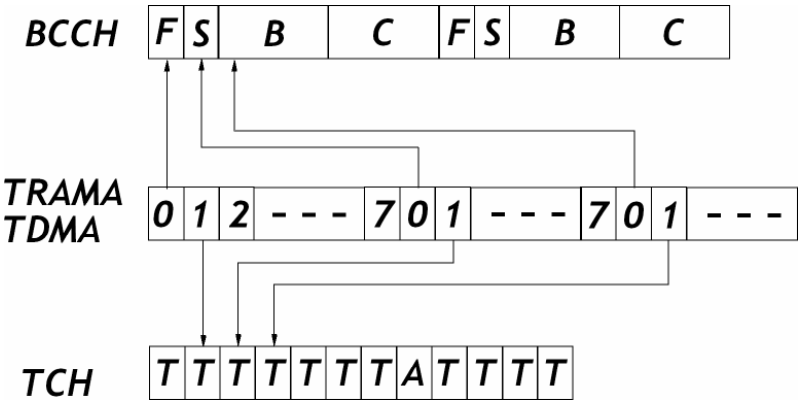


Figura 2.5.9 Multiplexado de canales lógicos en físicos. [20]

2.5.2.4. CONEXIONES A REDES E INTERFACES.

Entre cada par de elementos de la arquitectura GSM, existe una interfaz que requiere de su propio conjunto de protocolos para dar soporte a los datos de transito. La figura siguiente, ilustra todas las interfaces y señalizaciones utilizadas para la conexión entre los sistemas de una red GSM así como a redes de telefonía fija.

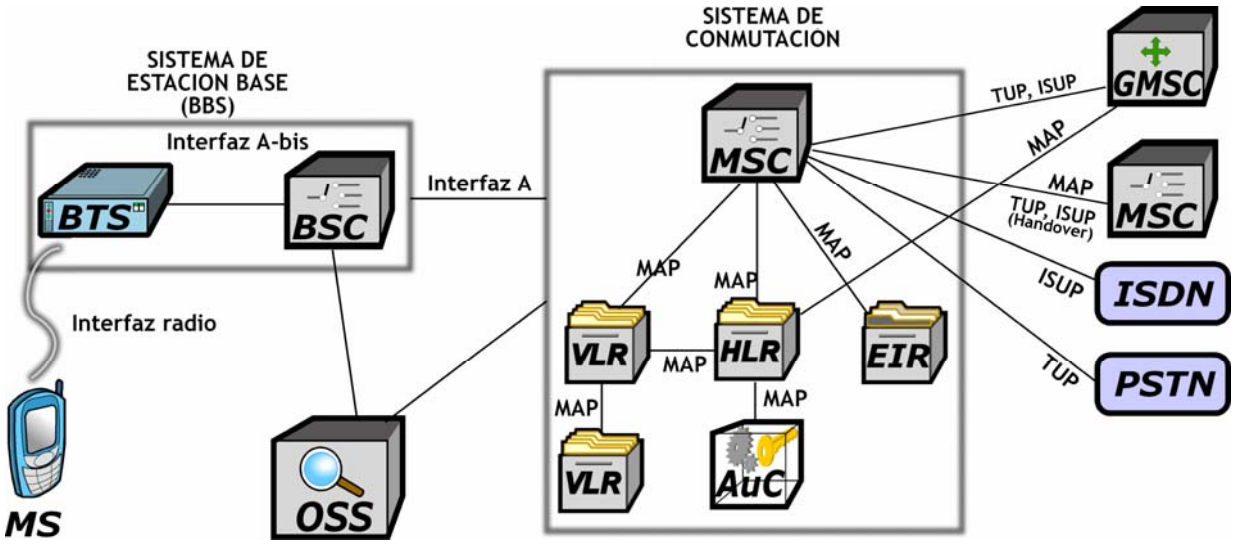


Figura 2.5.10 Interfaces y señalizaciones en una red GSM. [20]

2.5.2.4.1. REDES PSTN E ISDN.

Los sistemas de señalización usados entre una MSC y la red telefónica básica (PSTN) o la red digital de servicios integrados (ISDN) están individualmente diseñados para cumplir los requerimientos de las centrales a las cuales las MSC están conectadas. Los sistemas de señalización están basados en el sistema de señalización CCITT N°7 (señalización por canal común).

La MSC está integrada dentro de la red fija y tiene, para el establecimiento de llamadas, la misma interfaz que utilizan las centrales de la red fija.

La señalización hacia la red telefónica básica se realiza a través de la parte de usuario de telefonía (TUP), y hacia las centrales ISDN a través de la parte de usuario de servicios integrados (ISUP), todo ello dentro del sistema de señalización CCITT N° 7.

Para llamadas desde la red fija hacia un abonado móvil, la llamada tiene que ser enrutada hacia una Gateway MSC (GMSC), que efectúa la interrogación a los registros de posición para localizar al móvil. Cualquier MSC de la red de móviles puede asumir las funciones de GMSC.

2.5.2.4.2. SEÑALIZACIÓN DENTRO DEL SISTEMA DE CONMUTACIÓN.

La señalización dentro del sistema de conmutación de GSM está basada en las recomendaciones del CCITT para los sistemas de señalización por canal común (CCITT N°7).

Un nuevo esquema de señalización ha sido desarrollado especialmente para las redes GSM. La parte de aplicación de móviles (MAP), elaborada por el ETSI (Instituto Europeo de Estándares de Telecomunicación), corresponde a los niveles más altos de señalización entre la MSC y las siguientes entidades: los registros de posición (HLR y VLR), el AUC; el EIR y otras MSC. Los niveles de bajos de la señalización son gestionados por la parte de transferencia de mensajes (MTP) dentro del sistema de señalización CCITT N°7. Entre MAP y MTP se utilizan la parte de aplicación de las capacidades transaccionales (TCAP) y la parte de control de la conexión de señalización (SCCP).

La señalización debe operar internacionalmente entre las redes GSM y los registros de posición. Esta señalización es adicional al tráfico telefónico convencional, puesto que el GSM ha introducido nuevas características tales como el seguimiento internacional y la autentificación. Además de los dos mencionados, otros procedimientos gestionados por la MAP son:

- Traspaso de llamada (cambio de radiocanal en el transcurso de la llamada).
- Transferencia de información de abonado y actualización de la posición del abonado móvil en los registros de localización.

- Gestión de los servicios de abonado.
- Transferencia de datos de seguridad.

2.5.2.4.3. INTERFAZ A Y A-BIS.

La señalización entre el sistema de conmutación y el sistema de estaciones base (interfaz A), se realiza según la parte de aplicación del sistema de estaciones base (BSSAP) dentro del sistema de señalización CCITT N° 7. Algunos procedimientos gestionados por el BSSAP son:

- Asignación y liberación de recursos radio
- Traspaso de llamada
- Control de llamada
- Gestión de la movilidad, etc.

Los niveles bajos de señalización son gestionados por la parte de transferencia de mensajes (MTP) del sistema de señalización por canal común CCIYF N° 7.

La señalización entre BSC y BTS (interfaz A-bis) es implementada como un esquema especial de la señalización por canal común para canales PCM de 64 kbps. Uno de los canales se usa como canal de señalización transportando información de señalización según el protocolo de acceso de enlace sobre canal-D (LAPD) para GSM.

2.5.2.4.4. INTERFAZ RADIO.

La señalización entre la estación base y la estación móvil tiene lugar sobre la interfaz radio (también llamada interfaz Um).

La señalización usa un sistema específico de protocolo jerarquizado para GSM que utiliza los niveles 1, 2 y 3 del modelo OSI. Este protocolo se denomina Protocolo de Acceso de Enlace sobre Canal-Dm (LAPDm).

2.6. MODEMS CELULARES.

El modo de operación de un modem celular es un tanto parecido a un modem estándar utilizado para la transmisión de datos a través de líneas telefónicas, es decir, que éste se conecta a una computadora o host por medio de una interfaz serie y utilizando comandos AT se puede establece su configuración y posterior operación.

La diferencia radica en el uso de una antena para acceder a una red celular y poder transmitir la información, en lugar de una conexión física cableada a una línea telefónica analógica.

2.6.1. HISTORIA DE LOS MODEMS.

Los modems aparecieron como solución al problema de conectar computadoras remotas con el objetivo de transferir información sobre líneas telefónicas comunes. La palabra modem es una contracción de las palabras Modulador-Demodulador.

En el modem transmisor los datos modulan una señal portadora que es compatible con las señalizaciones utilizadas en las líneas telefónicas, y el modem receptor demodula la señal obteniendo los datos digitales.

Los primeros modems comerciales fueron desarrollados en 1,970. En el curso de estos años, los avances tecnológicos han permitido que los modems de marcación analógica, que inicialmente lograban enlaces de 300 bps, alcancen una velocidad

máxima de 56 kbps. En el campo de telefonía fija analógica, 56 kbps sea probablemente la velocidad tope que se puede alcanzar atravesando las PSTN[31].

Los modems inalámbricos por su parte, convierten las señales digitales de información a señales de radio bajo algún esquema de acceso a una red celular, y viceversa. A lo largo de 15 años de existencia, éstos modems han ofrecido velocidades de transmisión desde 9.6 kbps, hasta unos 153 kbps, conseguidos con los nuevos avances en el campo de la telefonía móvil.

En el mundo celular, GSM/GPRS y CDMA son dos estándares competidores con sus respectivas características y prestaciones, que se muestran en la tabla siguiente:

Aspectos	GSM/GPRS	CDMA2000 1xRTT
Conexiones	CSD[32] (GSM) o Conmutación de paquetes (GPRS)	CSD (IS95-B[33]) o conmutación de paquetes (1xRTT[34])
Velocidades ofrecidas	9.6 kbps – 85 kbps	9.6 kbps – 153 kbps
Soporte de SMS	Si	Si
Activación del servicio	Tarjeta SIM	Over-The-Air (OTA)
Modo de facturación	Por minuto (CSD) o por volumen transferido (GPRS)	Por minuto (CSD) o por volumen transferido (GPRS)
Acceso a redes IP	Si	Si
Cobertura	Mayoritariamente en áreas metropolitanas.	Mayoritariamente en áreas metropolitanas.

Tabla 2.6.1 Comparación de tecnologías. [22]

[31] PSTN: Public Switched Telephone Network.
[32] CSD: Circuit Switched Data.
[33] Estándar norteamericano para transferencia de datos CDMA.
[34] 1xRTT: 1 Channel of 1.25 MHz for Radio Transmission Technology.

2.6.2. ARQUITECTURA DE LAS CONEXIONES.

Existen tres modos de conexión para un modem celular: Envío de datos por conmutación de circuito (CSD), envío de datos por servicio de mensajería corta (SMS) y envío de datos por conmutación de paquetes. Los tres modos tienen características comunes con los modems de datos para líneas telefónicas.

Modem de marcación telefónica.	**Modem Inalámbrico**
Conexión de Modem a Modem.	Conmutación de circuitos.
Mensajería Instantánea.	Servicio de mensajería corta.
Conexión de Modem a Internet.	Conmutación de paquetes.

Tabla 2.6.2. Modos de conexión y su analogía con los modems tradicionales. [22]

2.6.2.1. ENVIO DE DATOS POR CONMUTACIÓN DE CIRCUITOS.

Las típicas conexiones punto a punto por conmutación de circuito, utilizando modems realizan la siguiente secuencia: se genera una llamada saliente desde el modem transmisor hacia el modem receptor utilizando la línea telefónica de la PSTN. La llamada es enrutada a través de las centrales telefónicas pertinentes hasta el modem solicitado, cerrando de esta forma el circuito. Durante la llamada, la línea enrutada se encuentra dedicada exclusivamente a la comunicación de los dos modems.

En una conexión de datos celular por conmutación de circuitos, la transmisión celular reemplaza la conexión telefónica de la PSTN. Por lo que la llamada saliente viaja por la infraestructura celular hasta el punto de interconexión o pasarela (Gateway) con la PSTN, que reside dentro de la red celular, a partir de allí el enrutamiento es idéntico al descrito anteriormente. La figura 2.6.1 ilustra un esquema de dicha conexión.

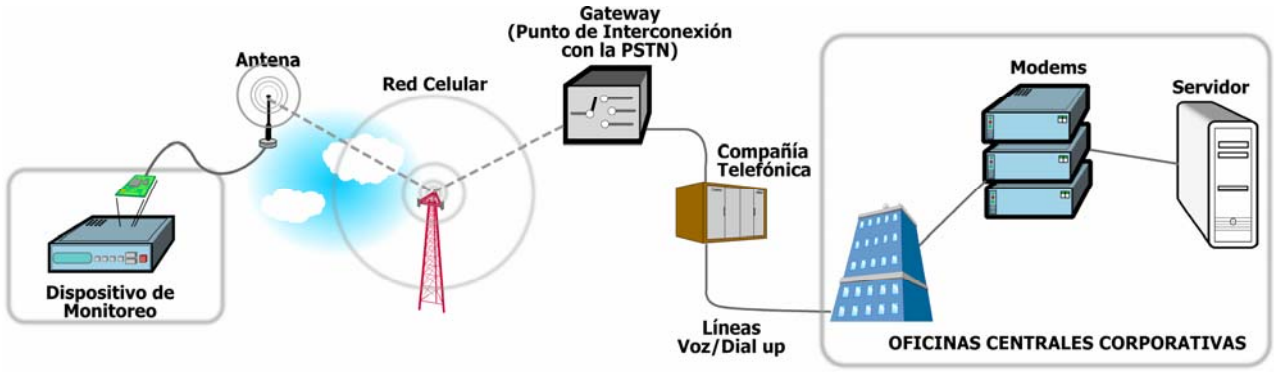


Figura 2.6.1 Configuración de una conexión celular CSD. [22]

Este tipo de arquitectura permite la conexión desde un punto de difícil acceso o móvil con casi cualquier terminación dentro del esquema de red operativo, con velocidades máximas de 14.4 kbps. Las conexiones celulares CSD son ideales en aplicaciones que requieren un reemplazo rápido con tecnología inalámbrica, de conexiones punto a punto existentes, ya que se requiere de pequeños cambios en la infraestructura.

2.6.2.2. ENVIO DE DATOS POR SERVICIO DE MENSAJERÍA CORTA.

El servicio de mensajería corta es análogo a la mensajería instantánea utilizada en Internet y a los sistemas de radio búsqueda (Pagers). Este sistema permite que se envíen mensajes de texto de hasta 160 caracteres hacia y desde dispositivos móviles. El sistema SMS cuenta con las mismas características de la telefonía celular, en cuanto al control de los traspasos de células e itinerancia. Si el dispositivo móvil se encuentra apagado o en áreas de cobertura nula, el servicio puede almacenar los mensajes y enviarlos una vez el dispositivo este activo.

Para dar soporte a este servicio, el operador de red celular debe contar con pasarelas a servidores SMS y/o a servidores de correo SMTP y con ello ofrecer el envío de mensajes entre móviles o desde un móvil a un buzón de correo y viceversa.

2.6.2.3. ENVIO DE DATOS POR CONMUTACIÓN DE PAQUETES.

Con algunas modificaciones, es posible lograr la transferencia de información por conmutación de paquetes sobre redes diseñadas para la conmutación de circuitos. La conmutación de paquetes es una técnica en donde la información (voz o datos) a ser transmitidos, es descompuesta en paquetes, de unos cuantos kilo bytes cada uno, los cuales son enrutados bajo direccionamientos IP. Con esto se logra un uso eficiente de los recursos de red, ya que solo se utiliza cuando es necesario el transporte de algún paquete.

En una conexión a Internet por modem tradicional, el modem establece una comunicación con el ISP[35], y éste permite el enlace (cambios en los formatos de la información) hacia redes IP. Una vez establecido el enlace, se pude llevar acabo transferencias en ambos sentidos utilizando los servicios que éstas redes ofrecen, tales como la transferencia de archivos por FTP, Navegación en la Web, Chat, e-mail, Telnet, etc.

La figura 2.6.2 ilustra que lo anterior descrito también puede ser llevado a cabo desde un dispositivo móvil, si la red de acceso a los recursos celulares lo permite (implica algunas modificaciones en la interfaz aire y en el sistema de estaciones base) y contando nuevamente con pasarelas para la interconexión hacia redes IP.

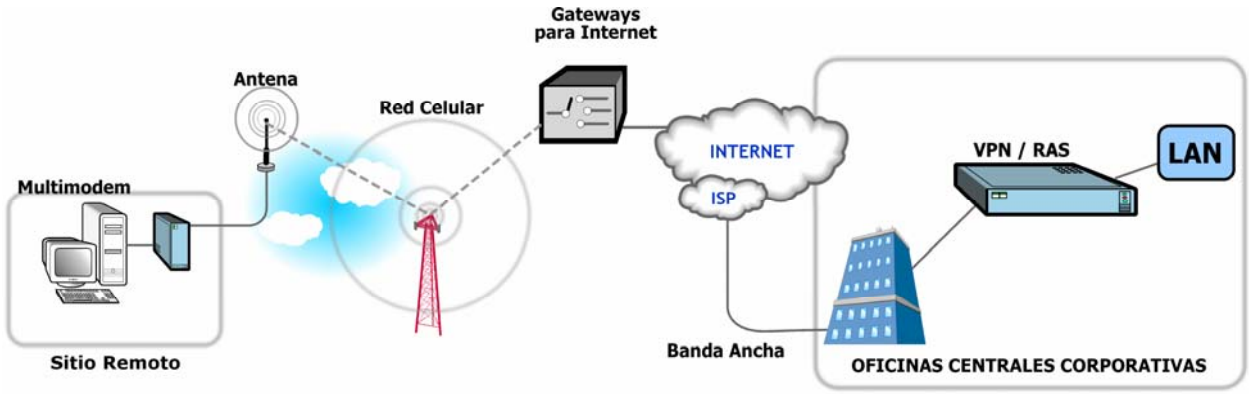


Figura 2.6.2 Configuración de una conexión celular por conmutación de paquetes. [22]

[35] ISP: Internet Service Provider.

CAPITULO III: *SOFTWARE*

3.1 JAVA 2 MICRO EDITION.

Debido a la expansión tecnológica y desde la aparición de la plataforma Java versión 2, se han orientado y desarrollado soluciones personalizadas para cada ámbito tecnológico. Sun Microsystems ha agrupado cada uno de esos ámbitos en una edición distinta de su lenguaje Java. Estas ediciones son Java 2 Standard Edition, orientada al desarrollo de aplicaciones independientes y de applets, Java 2 Enterprise Edition, enfocada al entorno empresarial y Java 2 Micro Edition, orientada a la programación de aplicaciones para pequeños dispositivos.

Si bien esta diversidad ofrecida por Sun Microsystems permite el desarrollo de aplicaciones que se ajustan a la capacidad de procesamiento de tres niveles distintos, todas las ediciones están basadas y comparten los aspectos genéricos heredados de la primera versión de Java. Por lo que, en el siguiente apartado se describe de forma breve ciertos fundamentos relacionados con ambas versiones.

3.1.1 ASPECTOS GENERALES DE JAVA.

La primera versión de Java fue desarrollada a principios de la década de los noventas por la empresa Sun Microsystems. Java desde su concepción, estaba pensado para ser un lenguaje de programación basado en la sintaxis de C, con las características de orientación a objetos de C++ y que además cumpliera con tres premisas: portabilidad, seguridad e independencia de la plataforma donde se ejecutase. Dadas estas características perseguidas, el ambiente de desarrollo apuntaba a la creación de software para el control de dispositivos electrónicos.

Paralelamente al desarrollo de Java, la World Wide Web tomaba impulso y ganaba terreno. De no haber sido contemporáneas ambas tecnologías, Java quizá hubiera sido

simplemente un lenguaje de utilidad para la programación de dispositivos electrónicos de consumo. La química entre ambos surge de la total compatibilidad en las premisas trazadas por los desarrolladores del nuevo lenguaje de programación.

En una red, entre el servidor y una computadora personal se transmiten dos tipos de objetos: la información pasiva, y la información dinámica, es decir, programas activos. Estos programas son agentes activos que se ejecuta en el host cliente, a pesar de que ha sido iniciado por el servidor. Aunque este tipo de programas ofrece grandes ventajas para el desarrollo de nuevas aplicaciones Web, también presenta algunos retos en las áreas de seguridad y portabilidad.

Básicamente Java se utiliza para crear aplicaciones y applets. Una aplicación es un programa que se ejecuta en el ordenador del usuario y bajo el sistema operativo de ese ordenador, es decir, la ejecución es similar a una aplicación desarrollada en cualquier otro lenguaje de programación.

Por otro lado, un applet es una aplicación diseñada para ser transmitida por Internet y ejecutada por un navegador compatible con Java, un applet es realmente un pequeño programa Java que se transfiere dinámicamente a través de la red, con la capacidad de reaccionar ante las acciones del usuario y cambiar dinámicamente.

3.1.1.1 SEGURIDAD Y PORTABILIDAD: EL BYTECODE.

La clave que permite a Java resolver los retos previstos en la seguridad y portabilidad, radica en la creación de un bytecode como respuesta de salida del compilador Java, en lugar de un código ejecutable. El bytecode es un conjunto de instrucciones altamente optimizado diseñado para ser ejecutado por una máquina virtual que emula al intérprete Java, nombrado Java Virtual Machine (JVM), es decir, el intérprete de Java es un intérprete de bytecode.

Traducir un programa Java a bytecode hace que su ejecución en una gran variedad de entornos resulte mucho más sencilla, y la razón es que para cada plataforma, sólo es necesario implementar el intérprete Java. Una vez que se dispone del programa de ejecución para un sistema determinado, cualquier programa Java puede ejecutarse en esa plataforma. Aunque algunos detalles del intérprete Java difieran de un sistema a otro, todos interpretan el mismo bytecode Java. Si Java fuera un lenguaje compilado, entonces deberían existir diferentes versiones del mismo programa para cada tipo de CPU conectada a Internet. Obviamente esta solución no es factible. Además, la interpretación del bytecode es la forma más sencilla de crear programas auténticamente portables.

El hecho de que Java sea interpretado también ayuda a hacerlo seguro. Como la ejecución de cada programa Java está bajo el control del intérprete Java, éste puede contener al programa e impedir que se generen efectos no deseados en el resto del sistema.

3.1.1.2 PRINCIPIOS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS BAJO EL ENFOQUE DE JAVA.

Los lenguajes de programación básicamente consisten en código y datos. Por otro lado, la organización conceptual de un programa puede girar en torno a su código o en torno a sus datos. La primera de estas dos formas se denomina modelo orientado al proceso. Este enfoque describe un programa como una serie de pasos lineales. Se puede considerar el modelo orientado al proceso como un código que actúa sobre los datos. Sin embargo, cuando se escriben programas bajo este enfoque, surgen problemas a medida que se hacen más largos y más complejos.

El segundo enfoque, llamado programación orientada a objetos fue concebido para solucionar esta complejidad. La programación orientada a objetos organiza un programa alrededor de sus datos, es decir, objetos, y de un conjunto de interfaces bien definidas con esos datos. Un programa orientado a objetos se puede delimitar

como un conjunto de datos que controlan el acceso al código. Bajo este enfoque se pueden conseguir muchas ventajas desde el punto de vista organizativo.

Todos los lenguajes orientados a objetos proporcionan tres mecanismos que ayudan a implementar el modelo. Estos mecanismos son el encapsulado, la herencia y el polimorfismo.

3.1.1.2.1 ENCAPSULADO.

El encapsulado es el mecanismo que permite unir el código junto con los datos que manipula, y mantiene a ambos a salvo de las interferencias exteriores y de un uso indebido. La potencia del código encapsulado es que su acceso y uso son sencillos sin necesidad de preocuparse por los detalles de la implementación, y sin temor a efectos inesperados.

En Java, la base del encapsulado es la *clase*. Una clase define la estructura y comportamiento (datos y código) que serán compartidos por un conjunto de objetos. Cada objeto de una determinada clase contiene la estructura y comportamiento definidos por la clase. Por este motivo, algunas veces se hace referencia a los objetos como instancias de una clase. Una clase es una construcción lógica, mientras que un objeto tiene una realidad física.

Cuando se crea una clase, se especifica el código y los datos que constituyen esa clase. En conjunto, estos elementos se denominan miembros de la clase. Los datos definidos por la clase se denominan variables de instancia. El código que opera sobre los datos se denomina *método*. Los métodos definen cómo se pueden utilizar las variables de instancia. Esto significa que el comportamiento y la interfaz de una clase están definidos por los métodos que operan sobre sus datos de instancia.

Dado que el objetivo de una clase es encapsular la complejidad, existen mecanismos para ocultar la complejidad de la implementación dentro de una clase. Cada método o variable dentro de una clase puede declararse como privada o pública. La interfaz pública de una clase representa todo lo que el usuario externo necesita o puede conocer. Solamente el código miembro de la clase puede acceder a métodos y datos privados. Por consiguiente, cualquier código que no sea miembro de la clase no tiene acceso a un método o variable privado. El acceso a miembros privados de una clase sólo puede hacerse a través de métodos públicos de clase; de esta forma se asegura que no ocurran acciones impropias. Esto significa que la interfaz pública debe ser diseñada cuidadosamente para no exponer demasiado los trabajos internos de una clase.

3.1.1.2.2 HERENCIA.

La herencia es el proceso por el cual un objeto adquiere las propiedades de otro. Esto es importante, ya que supone la base del concepto de clasificación jerárquica. Sin la utilización de jerarquías, cada objeto necesitaría definir explícitamente todas sus características. Sin embargo, mediante el uso de la herencia, un objeto sólo necesita definir aquellas cualidades que lo hacen único en su clase. Por lo tanto, el mecanismo de la herencia hace posible que un objeto sea una instancia específica de una clase más general.

La herencia interactúa también con el encapsulado. Si una determinada clase encapsula determinados atributos, entonces cualquier subclase tendrá los mismos atributos más cualquiera que añada como parte de su especialización. Una nueva subclase hereda todos los atributos de todos sus predecesores y no tiene interacciones con la mayoría del resto del código del sistema.

3.1.1.2.3 POLIMORFISMO.

El polimorfismo es una característica que permite que una interfaz sea utilizada por una clase general de acciones. La acción específica queda determinada por la naturaleza exacta de la situación.

De manera más general, el concepto de polimorfismo se puede resumir como una interfaz, múltiples métodos. Esto significa que es posible diseñar una interfaz genérica para un grupo de actividades relacionadas. Esto ayuda a reducir la complejidad permitiendo que la misma interfaz sea utilizada para especificar una clase general de acciones. Seleccionar la acción específica cuando se aplica a cada situación, es decir, el método, es tarea del compilador. El programador no necesita hacer esta selección manualmente sólo recordar y utilizar la interfaz general.

3.1.2 INTRODUCCIÓN A J2ME.

Java 2 Micro Edition fue presentada en 1999 por Sun Microsystems con el propósito de habilitar aplicaciones Java para pequeños dispositivos. En esta presentación, lo que realmente se enseñó fue una primera versión de una nueva JVM que podía ejecutarse en dispositivos de mano.

J2ME es la edición del lenguaje Java que está orientada al desarrollo de aplicaciones para dispositivos pequeños con capacidades restringidas tanto en pantalla gráfica, como de procesamiento y memoria, tales como teléfonos móviles, PDAs, Pagers, etc.

La aparición de esta tecnología esta ligada a las necesidades de los usuarios de telefonía móvil que cada vez demandan más servicios y prestaciones por parte tanto de los terminales como de las compañías. Además el uso de esta tecnología depende del asentamiento en el mercado de otras, como GPRS, íntimamente asociada a J2ME.

Actualmente las compañías telefónicas y los fabricantes de móviles están implantando los protocolos y cambios de infraestructura necesarios para darle soporte.

3.1.2.1 CONCEPTOS BÁSICOS DE J2ME.

Debido a que cada edición del paquete Java 2 está destinada a operar en un ambiente en particular, es necesario realizar una breve comparación entre estas, con el objetivo de delimitar las capacidades de cada una de ellas. A continuación se describen algunas características más sobresalientes de cada edición:

- Java 2 Platform, Standard Edition (J2SE): Esta edición de Java es la que en cierta forma recoge la iniciativa original del lenguaje Java, es decir, inspirada inicialmente en C++, pero con componentes de alto nivel, como soporte nativo de strings y recolector de basura. Código independiente de la plataforma, precompilado a bytecodes intermedio y ejecutado en el cliente por una JVM. Modelo de seguridad tipo *sandbox* proporcionado por la JVM. Abstracción del sistema operativo subyacente mediante un juego completo de APIs[36] de programación. Esta versión de Java contiene el conjunto básico de herramientas usadas para desarrollar Java Applets, así cómo las APIs orientadas a la programación de aplicaciones de usuario final: Interfaz gráfica de usuario, multimedia, redes de comunicación, etc.

- Java 2 Platform, Enterprise Edition (J2EE): La orientación de esta edición, es el entorno empresarial. El software empresarial tiene unas características propias marcadas: está pensado no para ser ejecutado en un equipo, sino para ejecutarse sobre una red de ordenadores de manera distribuida y remota mediante EJB's (Enterprise Java Beans). De hecho, el sistema se monta sobre varias unidades o aplicaciones. En muchos casos, además, el software empresarial requiere que se cuente con la capacidad de integrar datos provenientes de entornos heterogéneos. Esta edición está orientada especialmente al desarrollo de

[36] API: Application Programming Interface.

servicios Web, servicios de nombres, persistencia de objetos, XML[37], autenticación, APIs para la gestión de transacciones, etc. El cometido de esta especificación es ampliar la J2SE para dar soporte a los requisitos de las aplicaciones de empresa.

- Java 2 Platform, Micro Edition (J2ME): Esta edición está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades computacionales y gráficas muy reducidas, tales como teléfonos móviles, PDAs o electrodomésticos inteligentes. Esta edición tiene unos componentes básicos que la diferencian de las otras versiones, como el uso de una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en lugar de la clásica JVM, incluye también un pequeño y rápido recolector de basura y otras diferencias que se describen mas adelante.

La figura 3.1.1 presenta la arquitectura completa de la plataforma Java 2, la cual muestra un conjunto de tecnologías que abarcan todos los ámbitos de la computación.

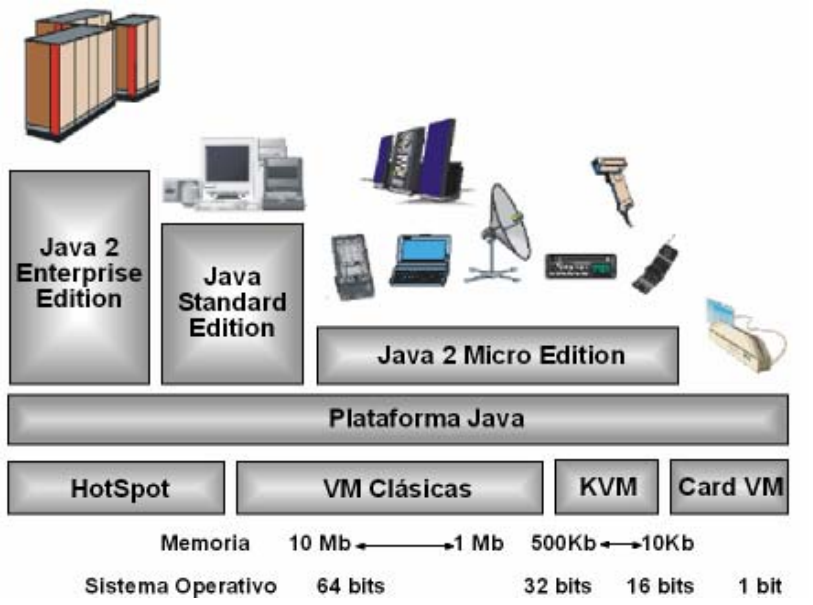


Figura 3.1.1 Arquitectura de la plataforma Java 2. [24]

[37] XML: Extensible Markup Language.

J2ME contiene un grupo reducido de las APIs de Java, debido a que la edición estándar de APIs de Java ocupa 20 Mb, y los dispositivos pequeños disponen de una cantidad de memoria mucho más reducida. Específicamente, J2ME usa 37 clases de la plataforma J2SE provenientes de los paquetes java.lang, java.io, java.util. Esta parte de la API que se mantiene fija forma parte de lo que se denomina *configuración*.

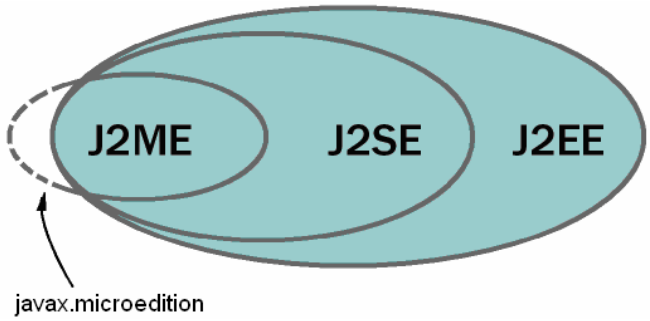


Figura 3.1.2 Relación entre las APIs de la plataforma Java. [24]

Como se observa en la figura 3.1.2, J2ME representa una versión simplificada de J2SE. La separación en tres ediciones distintas responde a razones de eficiencia. Por lo tanto, J2EE es un súper conjunto de J2SE pues contiene toda la funcionalidad de éste y más características relacionadas con E/S y entornos de red. De forma similar J2ME es un subconjunto de J2SE (excepto por el paquete javax.microedition), el cual esta relacionado con las restricciones del ambiente de operación de J2ME.

La tecnología de J2ME se divide en tres componentes o capas principales, teniendo cada una la tarea de brindar soporte o servicio a la capa superior, estos componentes, en orden ascendente, son:

- Maquinas Virtuales con diferentes requisitos y prestaciones para adaptarse a los distintos tipos de pequeños dispositivos.

- Las configuraciones, que son un conjunto de clases básicas orientadas a conformar el corazón de las implementaciones para dispositivos de características específicas. Existen dos configuraciones definidas en J2ME:

CLDC (Connected Limited Device Configuration) enfocada a dispositivos con restricciones de procesamiento y memoria, y la llamada CDC (Connected Device Configuration) enfocada a dispositivos con más recursos.

- Perfiles, que son bibliotecas Java de clases específicas orientadas a implementar funcionalidades de más alto nivel para familias específicas de dispositivos.

La arquitectura de un entorno de ejecución típico para J2ME se puede observar en la figura 3.1.3, en ésta se presentan los tres componentes de la tecnología montados sobre el sistema operativo compatible del dispositivo, encargado de enlazar el hardware y software para un correcto funcionamiento.

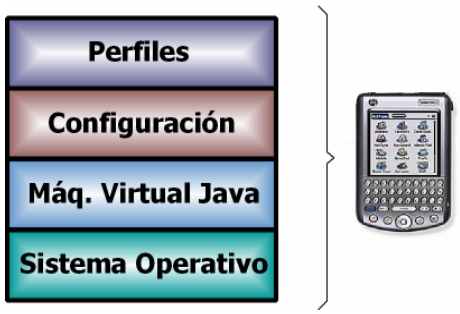


Figura 3.1.3 Entorno de ejecución. [24]

3.1.2.2 LA MÁQUINA VIRTUAL DE J2ME.

Tal como describió anteriormente, una máquina virtual de Java (JVM) es un programa encargado de interpretar código intermedio (bytecode) de los programas Java precompilados a código máquina ejecutable por la plataforma, efectuar las llamadas pertinentes al sistema operativo subyacente y observar las reglas de seguridad y corrección de código definidas para el lenguaje Java.

De esta forma, la JVM proporciona al programa Java independencia de la plataforma con respecto al hardware y al sistema operativo sobre el cual opera. Las

implementaciones tradicionales de JVM son, en general, muy pesadas en cuanto a la memoria ocupada y requerimientos computacionales.

J2ME define varias JVMs de referencia adecuadas al ámbito de los dispositivos electrónicos que, en algunos casos, suprimen algunas características con el fin de obtener una implementación menos exigente.

Ya que máquina virtual proporciona soporte a las configuraciones, se requiere una máquina virtual para cada tipo. La VM (Virtual Machine) de la configuración CLDC se denomina KVM y la relacionada con la configuración CDC se denomina CVM.

3.1.2.2.1 KILO VIRTUAL MACHINE.

Es la máquina virtual más pequeña desarrollada por Sun Microsystems. Su nombre KVM proviene de kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40 Kb y 80 Kb). Se trata de una implementación de máquina virtual reducida y especialmente orientada a dispositivos con bajas capacidades computacionales y de memoria. La KVM está escrita en lenguaje C, aproximadamente unas 24000 líneas de código, y fue diseñada para ser:

- Pequeña, con una carga de memoria entre los 40 Kb y los 80 Kb, dependiendo de la plataforma y las opciones de compilación.
- Alta portabilidad.
- Modulable.
- Lo más completa y rápida posible y sin sacrificar características para las que fue diseñada.

Sin embargo, esta baja ocupación de memoria hace que posea algunas limitaciones con respecto a la clásica Java Virtual Machine (JVM):

- No hay soporte para tipos en coma flotante. No existen por tanto los tipos double ni float. Esta limitación está presente porque los dispositivos carecen del hardware necesario para estas operaciones.
- No existe soporte para JNI (Java Native Interface) debido a los recursos limitados de memoria.
- No existen cargadores de clases (class loaders) definidos por el usuario. Sólo existen los predefinidos.
- No se permiten los grupos de hilos o hilos daemon. En lugar de ello, se utilizan los objetos *Colección* para almacenar cada hilo en el ámbito de la aplicación.
- No existe la finalización de instancias de clases. No existe el método Object.finalize().
- No hay referencias débiles[38].
- Limitada capacidad para el manejo de excepciones debido a que el manejo de éstas depende en gran parte de las APIs de cada dispositivo por lo que son éstos los que controlan la mayoría de las excepciones.
- Reflexión[39].

Aparte de la no inclusión de estas características, la verificación de clases merece un comentario aparte. El verificador de clases estándar de Java es demasiado grande para la KVM. De hecho es más grande que la propia KVM y el consumo de memoria es excesivo, más de 100 Kb para las aplicaciones típicas.

Este verificador de clases es el encargado de rechazar las clases no válidas en tiempo de ejecución. Este mecanismo verifica los *bytecodes* de las clases Java realizando las siguientes comprobaciones:

[38] Un objeto que está siendo apuntado mediante una referencia débil es un candidato para la recolección de basura. Estas referencias están permitidas en J2SE, pero no en J2ME.

[39]La reflexión es el mecanismo por el cual los objetos pueden obtener información de otros objetos en tiempo de ejecución tales como los archivos de clase cargados o sus campos y métodos.

- Ver que el código no sobrepase los límites de la pila de la VM.
- Comprobar que no se utilizan las variables locales antes de ser inicializadas.
- Comprobar que se respetan los campos, métodos y los modificadores de control de acceso a clases.

Por esta razón los dispositivos que usen la configuración CLDC y KVM introducen un algoritmo de verificación de clases en dos pasos. Este proceso puede apreciarse gráficamente en la Figura 3.1.4.

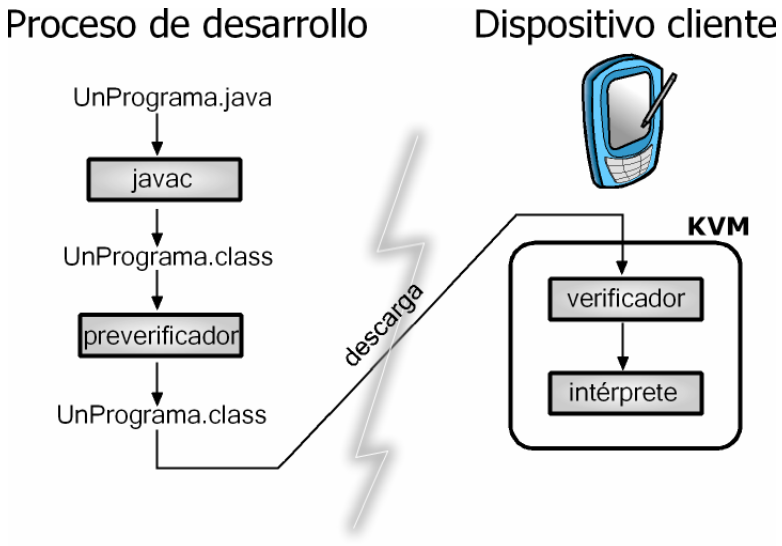


Figura 3.1.4 Preverificación de clases en CDLC/KVM. [24]

3.1.2.2.2 COMPACT VIRTUAL MACHINE.

La CVM (Compact Virtual Machine) ha sido tomada como Máquina Virtual Java de referencia para la configuración CDC y soporta las mismas características que la Máquina Virtual de J2SE. Está orientada a dispositivos electrónicos con procesadores de 32 bits de gama alta y en torno a 2Mb o más de memoria RAM. Las características que presenta esta Máquina Virtual son:

- Sistema de memoria avanzado.
- Tiempo de espera bajo para el recolector de basura.

- Separación completa de la VM del sistema de memoria.
- Recolector de basura modularizado.
- Portabilidad.
- Rápida sincronización.
- Ejecución de las clases Java fuera de la memoria de sólo lectura (ROM).
- Soporte nativo de hilos.
- Baja ocupación en memoria de las clases.
- Proporciona soporte e interfaces para servicios en Sistemas Operativos de Tiempo Real.
- Conversión de hilos Java a hilos nativos.
- Soporte para todas las características de Java2 y librerías de seguridad, referencias débiles, Interfaz Nativa de Java (JNI), invocación remota de métodos (RMI), Interfaz de depuración de la Máquina Virtual (JVMDI).

3.1.2.3 CONFIGURACIONES.

Una configuración es el conjunto mínimo de APIs Java que permiten desarrollar aplicaciones para un grupo de dispositivos. Éstas APIs describen las características básicas, comunes a todos los dispositivos:

- Características soportadas del lenguaje de programación Java.
- Características soportadas por la Máquina Virtual Java.
- Bibliotecas básicas de Java y APIs soportadas.

Existen dos configuraciones en J2ME: CLDC, orientada a dispositivos con limitaciones computacionales y de memoria y CDC, orientada a dispositivos con no tantas limitaciones.

3.1.2.3.1 CONNECTED DEVICE CONFIGURATION (CDC).

La CDC está orientada a dispositivos con cierta capacidad computacional y de memoria. Por ejemplo, decodificadores de televisión digital, televisores con internet, algunos electrodomésticos y sistemas de navegación en automóviles. CDC usa una Máquina Virtual Java (CVM) similar en sus características a una de J2SE, pero con limitaciones en el apartado gráfico y de memoria del dispositivo. La CDC está enfocada a dispositivos con las siguientes capacidades:

- Procesador de 32 bits.
- Disponer de 2 Mb o más de memoria total, incluyendo memoria RAM y ROM.
- Poseer la funcionalidad completa de la Máquina Virtual Java2.
- Conectividad a algún tipo de red.

La CDC está basada en J2SE e incluye varios paquetes Java de la edición estándar. Las peculiaridades de la CDC están contenidas principalmente en el paquete javax.microedition.io, que incluye soporte para comunicaciones HTTP[40] y basadas en datagramas. La Tabla 3.1.1 presenta las librerías incluidas en la CDC.

Nombre de Paquete CDC	Descripción
java.io	Clases e interfaces estándar de E/S.
java.lang	Clases básicas de lenguaje.
java.lang.ref	Clases de referencia.
java.lang.reflect	Clases e interfaces de reflexión.
java.math	Paquete de matemáticas.
java.net	Clases e interfaces de red.
java.security	Clases e interfaces de seguridad.
java.security.cert	Clases de certificados de seguridad.
java.text	Paquete de texto.
java.util	Clases de utilidades estándar.
java.util.jav	Clases y utilidades para archivos JAR.
java.util.zip	Clases y utilidades para archivos ZIP y comprimidos.
javax.microedition.io	Clases e interfaces para conexión genérica CDC.

Tabla 3.1.1 Librerías de configuración CDC. [24]

[40] HTTP: Hyper Text Transfer Protocol.

3.1.2.3.2 CONNECTED LIMITED DEVICE CONFIGURATION (CDLC).

La CLDC está orientada a dispositivos dotados de conexión y con limitaciones en cuanto a capacidad gráfica, cómputo y memoria. Ejemplos de estos dispositivos son: teléfonos móviles, pagers y PDAs. Los dispositivos que usan CLDC deben cumplir los siguientes requisitos:

- Requisitos de Hardware. Las capacidades de hardware de los dispositivos que soportan CLCD varían considerablemente, por lo que se imponen los siguientes requisitos mínimos:
  - El dispositivo debe disponer entre 160 Kb y 512 Kb de memoria total. Como mínimo contar con 128 Kb de memoria no volátil para la Máquina Virtual Java y las bibliotecas CLDC, y 32 Kb de memoria volátil para la Máquina Virtual en tiempo de ejecución.
  - Procesador de 16 o 32 bits con al menos 25 Mhz de velocidad.
  - Ofrecer bajo consumo de energía, debido a que estos dispositivos trabajan con suministros energéticos limitados, normalmente baterías.
  - Tener conexión a algún tipo de red, normalmente sin cable, con conexión intermitente y ancho de banda limitado (unos 9600 bps).

- Requisitos de Software. Generalmente, la configuración CLDC asume que el dispositivo contiene un mínimo Sistema Operativo encargado del manejo del hardware de éste. Pero al igual que las capacidades hardware, el software incluido en los dispositivos CLDC varía considerablemente. Dentro de esta variedad, CLDC define unas mínimas características que deben poseer el software de los dispositivos CLDC:
  - Este Sistema Operativo debe proporcionar al menos una entidad de planificación para ejecutar la JVM.
  - El Sistema Operativo no necesita soportar espacios de memoria separados o procesos, ni debe garantizar la planificación de procesos en tiempo real o comportamiento latente.

La CLDC aporta las siguientes funcionalidades a los dispositivos:

- Un subconjunto del lenguaje Java y todas las restricciones de su Máquina Virtual (KVM).
- Un subconjunto de las bibliotecas Java del núcleo.
- Soporte para E/S básica.
- Soporte para acceso a redes.
- Seguridad. Esta es importante ya que los dispositivos CLDC almacenan información personal del usuario, por lo que es necesario asegurar la integridad de los datos transmitidos y de las aplicaciones. Para ello se lleva a cabo un modelo similar a los sandbox que operan durante la ejecución de un applet. Este modelo solicita a las aplicaciones el cumplimiento de las siguientes condiciones:

  - Los archivos de clase Java deben ser verificados como aplicaciones Java válidas.
  - Sólo se permite el uso de APIs autorizadas por el CLCD.
  - No está permitido cargar clases definidas por el usuario.
  - Sólo se puede acceder a características nativas que estén dentro del CLCD.
  - Una aplicación ejecutada bajo KVM no debe ser capaz de dañar el dispositivo dónde se encuentra. De esto se encarga el verificador de clases que se asegura que no haya referencias a posiciones no válidas de memoria. También comprueba que las clases cargadas no se ejecuten de una manera no permitida por las especificaciones de la máquina virtual.

CLDC proporciona un conjunto de clases heredadas de la plataforma J2SE. En total, son 37 clases provenientes de los paquetes java.lang, java.util y java.io. Cada una de estas clases debe ser idéntica o ser un subconjunto de la correspondiente clase de J2SE. Tanto los métodos como la semántica de cada clase deben permanecer invariables. La tabla 3.1.2 muestra estas clases heredadas.

Nombre de paquete CLCD	Descripción	Clases
java.io	Clases y paquetes estándar de E/S.	java.io.ByteArrayInputStream
		java.io.ByteArrayOutputStream
		java.io.DataInput
		java.io.DataOutput
		java.io.DataInputStream
		java.io.DataOutputStream
		java.io.InputStream
		java.io.InputStreamReader
		java.io.OutputStream
		java.io.OutputStreamWriter
		java.io.PrintStream
		java.io.Reader
java.lang	Clases y paquetes de la máquina virtual	java.lang.Class
		java.lang.Object
		java.lang.Runnable
		java.lang.Runtime
		java.lang.String
		java.lang.Stringbuffer
		java.lang.System
		java.lang.Thread
		java.lang.Throwable
	Clases de datos	java.lang.Boolean
		java.lang.Byte
		java.lang.Character
		java.lang.Integer
		java.lang.Long
		java.lang.Short
java.util	Clases, interfaces y utilidades estándar.	java.util.Calendar
		java.util.Date
		java.util.Enumeration
		java.util.Hashtable
		java.util.Random
		java.util.Stack
		java.util.TimeZone
		java.util.Vector

Tabla 3.1.2 Clases heredadas de J2SE. [24]

La plataforma J2SE contiene a los paquetes java.io y java.net encargados de las operaciones de E/S. Debido a las limitaciones de memoria de CLDC no es posible la inclusión de todas las clases de estos paquetes. CLDC hereda algunas clases del

paquete java.io, pero no hereda ninguna clase relacionada con la E/S de mayor envergadura. Esto es debido a la gran variedad de dispositivos que abarca CLDC, ya que, para éstos puede resultar innecesario manejar la transferencia de archivos. No se han incluido tampoco las clases del paquete java.net, basado en comunicaciones TCP/IP ya que no todos los dispositivos CLDC tienen que basarse en este protocolo de comunicación. Para suplir estas carencias CLDC posee un conjunto de clases más genérico para la E/S y la conexión a red recibe el nombre de "Generic Connection Framework". Estas clases están incluidas en el paquete javax.microedition.io y son las que aparecen en la Tabla 3.1.3.

Clase	Descripción
Connector	Clase genérica que puede crear cualquier tipo de conexión.
Connection	Interfaz que define el tipo de conexión más genérica.
InputConnection	Interfaz que define una conexión de streams de entrada.
OutputConnection	Interfaz que define una conexión de streams de salida.
StreamConnection	Interfaz que define una conexión basada en streams.
ContentConnection	Extensión a StreamConnection para trabajar con datos.
Datagram	Interfaz genérico de datagramas.
DatagramConnection	Interfaz que define una conexión basada en datagramas.
StreamConnectionNotifier	Interfaz que notifica una conexión. Permite crear una conexión en el lado del servidor.

Tabla 3.1.3 Clases e interfaces incluidos en el paquete javax.microedition.io [24]

3.1.2.4 PERFILES.

Un perfil es un conjunto de APIs orientado a un ámbito de aplicación determinado. Los perfiles identifican un grupo de dispositivos por la funcionalidad que proporcionan y el tipo de aplicaciones que se ejecutarán en ellos. Las librerías de la interfaz gráfica son un componente muy importante en la definición de un perfil. Aquí se encuentran grandes diferencias entre interfaces, desde el menú textual de los teléfonos móviles hasta los táctiles de los PDAs.

El perfil establece unas APIs que definen las características de un dispositivo, mientras que la configuración hace lo propio con una familia de ellos. Esto hace que a la hora de construir una aplicación se cuente tanto con las APIs del perfil como de

la configuración. Un perfil siempre se construye sobre una configuración determinada y por lo tanto existirán perfiles para cada tipo de configuración. En la figura 3.1.5 se ilustra el entorno de ejecución completo así como los perfiles definidos para cada configuración.

Figura 3.1.5 Arquitectura del entorno de ejecución de J2ME. [24]

- Foundation Profile: Este perfil define una serie de APIs sobre la CDC orientadas a dispositivos que carecen de interfaz gráfica Este perfil incluye gran parte de los paquetes de la J2SE, pero excluye totalmente los paquetes java.awt y java.swing que conforman la GUI[41] de J2SE. Si una aplicación requiriera una GUI, entonces sería necesario un perfil adicional. Los paquetes que forman parte del Foundation Profile se muestran en la Tabla 3.1.4.

Paquete del Foundation Profile	Descripción
Java.lang	Soporte del lenguaje Java
Java.util	Añade soporte completo para ZIP y otras funcionalidades (java.util.Timer)
Java.net	Incluye sockets TCP/IP y conexiones HTTP
Java.io	Clases Reader y Writer de J2SE
Java.text	Incluye soporte para internacionalización
Java.security	Incluye códigos y certificados.

Tabla 3.1.4 Librerías del Foundation Profile. [24]

[41] GUI: Guide Users Interface.

- El Personal Profile es un subconjunto de la plataforma J2SE, y proporciona un entorno con un completo soporte gráfico AWT[42]. El objetivo es el de dotar a la configuración CDC de una interfaz gráfica completa, con capacidades Web y soporte de applets Java. Este perfil requiere una implementación del Foundation Profile. La Tabla 3.1.5 presenta los paquetes que conforman el Personal Profile v1.0.

Paquete del Personal Profile	Descripción
java.applet	Clase necesitada para la creación de applets o usadas por ellos.
java.awt	Clases para crear GUIs con AWT
java.awt.datatransfer	Clases e interfaces para transmitir datos entre aplicaciones.
java.awt.event	Clases e interfaces para manejar eventos AWT.
java.awt.font	Clases e interfaces para la manipulación de fuentes.
Java.awt.im	Clases e interfaces para definir métodos editores de entrada.
Java.awt.im.spi	Interfaces que añaden el desarrollo de métodos editores de entrada para cualquier entorno de ejecución Java.
Java.awt.image	Clases para crear y modificar imágenes.
Java.beans	Clases que soportan JavaBeans.
Javax.microedition.xlet	Interfaces que usa el Personal Profile para la comunicación.

Tabla 3.1.5 Librerías del Personal Profile. [24]

- RMI[43] Profile. Este perfil requiere una implementación del Foundation Profile. El perfil RMI soporta un subconjunto de las APIs J2SE RMI. Algunas características de estas APIs se han eliminado del perfil RMI debido a las limitaciones de cómputo y memoria de los dispositivos. Las siguientes propiedades se han eliminado del J2SE RMI:
  - Java.rmi.server.disableHTTP.
  - Java.rmi.activation.port.
  - Java.rmi.loader.packagePrefix.
  - Java.rmi.registry.packagePrefix.
  - Java.rmi.server.packagePrefix.

[42] AWT: Abstract Window Toolkit.
[43] RMI: Remote Method Invocation.

- PDA Profile: Está construido sobre CLDC. Pretende abarcar PDAs de gama baja, con una pantalla y algún tipo de puntero y una resolución de al menos 20000 pixels (al menos 200x100 pixels) con un factor 2:1. Este perfil se encuentra en fase de definición.

- MIDP (Mobile Information Device Profile): Este perfil está construido sobre la configuración CLDC. Al igual que CLDC fue la primera configuración definida para J2ME, MIDP fue el primer perfil definido para esta plataforma. Este perfil está orientado para dispositivos con las siguientes características:
  - Reducida capacidad computacional y de memoria.
  - Conectividad limitada.
  - Capacidad gráfica muy reducida.
  - Entrada de datos alfanumérica reducida.
  - 128 Kb de memoria no volátil para componentes MIDP.
  - 8 Kb de memoria no volátil para datos persistentes de aplicaciones.
  - 32 Kb de memoria volátil en tiempo de ejecución para la pila Java.

El perfil MIDP establece las capacidades del dispositivo, por lo tanto, especifica las APIs relacionadas con:

- La aplicación (semántica y control de la aplicación MIDP).
- Interfaz de usuario.
- Almacenamiento persistente RMS[44].
- Trabajo en red.

Paquetes del MIDP	Descripción
javax.microedition.lcdui	Clases e interfaces para GUIs
javax.microedition.rms	Soporte para el almacenamiento persistente del dispositivo.
javax.microedition.midlet	Clases de definición de la aplicación.
javax.microedition.io	Clases e interfaces de conexión genérica.
javax.io	Clases e interfaces de E/S básica.
javax.lang	Clases e interfaces de la máquina virtual.
javax.util	Clases e interfaces de utilidades estándar.

Tabla 3.6 Librerías del perfil MIDP. [24]

[44] RMS: Record Management Storage.

3.1.2.5 LOS MIDLETS.

Las aplicaciones realizadas con J2ME (MIDlets si están desarrolladas bajo especificaciones MIDP) están pensadas para que puedan ser descargadas a través de una conexión a Internet. El medio empleado para garantizar esta descarga recibe el nombre de OTA (Over the Air). Aunque dependiendo de las capacidades de conectividad local del dispositivo, estas se pueden descargar desde una computadora.

Una aplicación J2ME está formada por un archivo JAR (Java Archive) que contiene a la aplicación en sí y un archivo JAD (Java Archive Descriptor) que contiene diversa información sobre la aplicación.

3.1.2.5.1 EL GESTOR DE APLICACIONES.

El gestor de aplicaciones o AMS (Application Management System) es el software encargado de gestionar los MIDlets. Este software reside en el dispositivo y es el que permite ejecutar, pausar o destruir las aplicaciones J2ME. El AMS realiza dos grandes funciones: gestiona el ciclo de vida de los MIDlets, y se encarga de controlar los estados por los que pasa el MIDlet mientras está en la memoria del dispositivo, es decir, en ejecución.

3.1.2.5.1.1 CICLO DE VIDA DE UN MIDLET.

El ciclo de vida de un MIDlet pasa por cinco fases: localización o descubrimiento, instalación, ejecución, actualización y borrado. Ver figura 3.1.6. El AMS es el encargado de gestionar cada una de estas fases de la siguiente manera:

- Localización: En esta fase el gestor de aplicaciones proporciona los mecanismos necesarios para realizar la elección del MIDlet a descargar. El AMS puede ser capaz de realizar la descarga de aplicaciones de diferentes maneras, dependiendo de las capacidades del dispositivo, es decir, esta descarga se puede realizar mediante un cable conectado a una PC o mediante una conexión inalámbrica.

- Instalación: Una vez descargado el MIDlet en el dispositivo, comienza el proceso de instalación. En esta fase el gestor de aplicaciones controla todo el proceso informando al usuario tanto de la evolución de la instalación como de si existiese algún problema durante ésta. Cuando un MIDlet está instalado en el dispositivo, todas sus clases, archivos y almacenamiento persistente están preparados y listos para su uso.

- Ejecución: En esta fase, el AMS tiene la función de gestionar los estados del MIDlet en función de los eventos que se produzcan durante esta ejecución.

- Actualización: El AMS debe ser capaz de detectar después de una descarga si el MIDlet descargado es una actualización de un MIDlet ya presente en el dispositivo. Si es así, debe informar de ello, y permitir si se realizará la actualización pertinente o no.

- Borrado: En esta fase el AMS es el encargado de borrar el MIDlet seleccionado del dispositivo. El AMS pedirá confirmación antes de proceder a su borrado e informará de cualquier circunstancia que se produzca.

Un MIDlet puede permanecer en el dispositivo durante un tiempo indefinido. Después de la fase de instalación, el MIDlet queda almacenado en una zona de memoria persistente del dispositivo MID. El usuario de éste dispositivo es el encargado de decidir en qué momento quiere eliminar la

aplicación y así se lo hará saber al AMS mediante alguna opción que éste suministre.

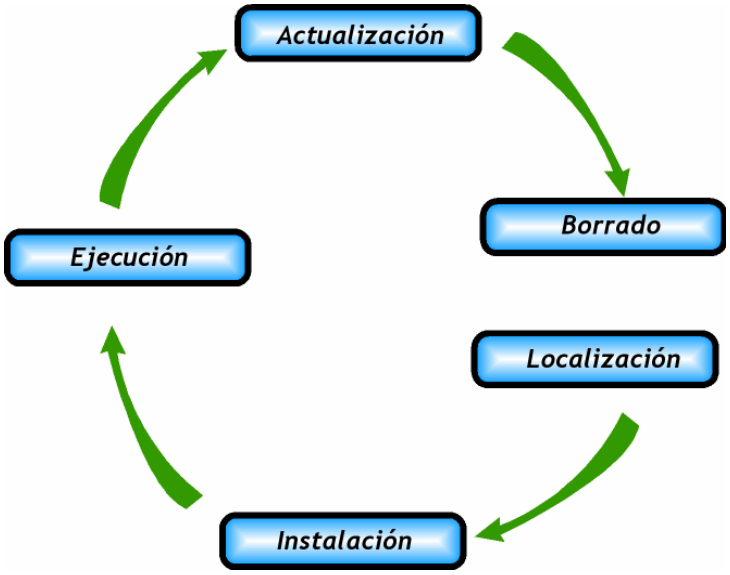


Figura 3.1.6 Ciclo de vida de un MIDlet. [24]

3.1.2.5.1.2 ESTADOS DE UN MIDLET EN FASE DE EJECUCIÓN.

Además de gestionar el ciclo de vida de los MIDlets, el AMS es el encargado de controlar los estados del MIDlet durante su ejecución. Mientras exista el MIDlet éste se encuentra cargado en la memoria del dispositivo y es aquí donde puede transitar entre tres estados diferentes: activo, en pausa y destruido.

Cuándo un MIDlet comienza su ejecución, se encuentra en el estado Activo. El gestor de aplicaciones debe ser capaz de cambiar el estado de la aplicación, en función a eventos externos al ámbito de ejecución que puedan producir, tales como llamadas entrantes o la recepción de mensajes.

En este caso, el gestor de aplicaciones interrumpirá la ejecución del MIDlet sin que se viese afectada la ejecución de éste y lo ubicará en el estado de Pausa para atender la llamada o leer el mensaje.

Una vez finalizada la ejecución del MIDlet, éste pasará al estado de Destruido dónde sería eliminado de la memoria del dispositivo. Cuando se dice que el MIDlet pasa al estado Destruido y es eliminado de la memoria, se refiere a la memoria volátil del dispositivo que es usada para la ejecución de aplicaciones. Una vez finalizada la ejecución del MIDlet se puede volver a invocarlo las veces que se deseen ya que éste permanece en la zona de memoria persistente hasta el momento de su desinstalación.

La Figura 3.1.7 ilustra el diagrama de estados de un MIDlet en ejecución:

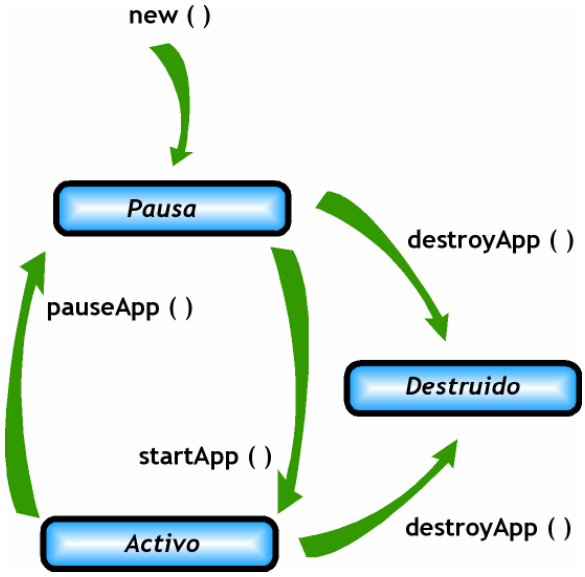


Figura 3.1.7 Estados de un MIDlet en ejecución. [24]

Como se observa en el diagrama, un MIDlet puede cambiar de estado mediante una llamada a los métodos MIDlet.startApp( ), MIDlet.pauseApp( ) o MIDlet.destroyApp( ). El gestor de aplicaciones cambia el estado de los MIDlets haciendo una llamada a cualquiera de los métodos anteriores. Un MIDlet también puede cambiar de estado por sí mismo.

En primer lugar, se realiza la llamada al constructor del MIDlet pasando éste al estado de Pausa durante un corto período de tiempo. El AMS por su parte crea una nueva instancia del MIDlet. Cuándo el dispositivo está preparado para ejecutar el MIDlet, el AMS invoca al método MIDlet.startApp( ) para entrar en el estado de Activo. El MIDlet entonces, ocupa todos los recursos que necesita para su ejecución. Durante este estado, el MIDlet puede pasar al estado de Pausa por una acción del usuario, o bien, por el AMS que reduciría en todo lo posible el uso de los recursos del dispositivo por parte del MIDlet.

Tanto en el estado Activo como en el de Pausa, el MIDlet puede pasar al estado Destruido realizando una llamada al método MIDlet.destroyApp(). Esto puede ocurrir porque el MIDlet haya finalizado su ejecución o porque una aplicación prioritaria necesite ser ejecutada en memoria en lugar del MIDlet. Una vez destruido el MIDlet, éste libera todos los recursos ocupados.

3.2 BASE DE DATOS.

3.2.1 NOCIONES DE BASE DE DATOS RELACIONALES.

Las bases de datos permiten recolectar todo tipo de información con fines de almacenamiento, búsqueda y posterior recuperación de la misma. Por otra parte, una base de datos relacional es un conjunto de tablas relacionadas entre sí; cada tabla está definida por una serie de campos y conformada por una lista de tuplas. La figura 3.2.1 muestra un ejemplo explicativo de estos conceptos.

Los campos forman las columnas de las tablas, definen el tipo y variedad de sus datos. Las filas de datos se denominan tuplas o registros; una tabla puede estar vacía (sin ninguna tupla) o contener un número variable de las mismas. A cada valor de un

campo definido en una tupla, se le denomina atributo. Cada tabla de una base de datos puede contener un número de tuplas diferente al de las demás tablas.

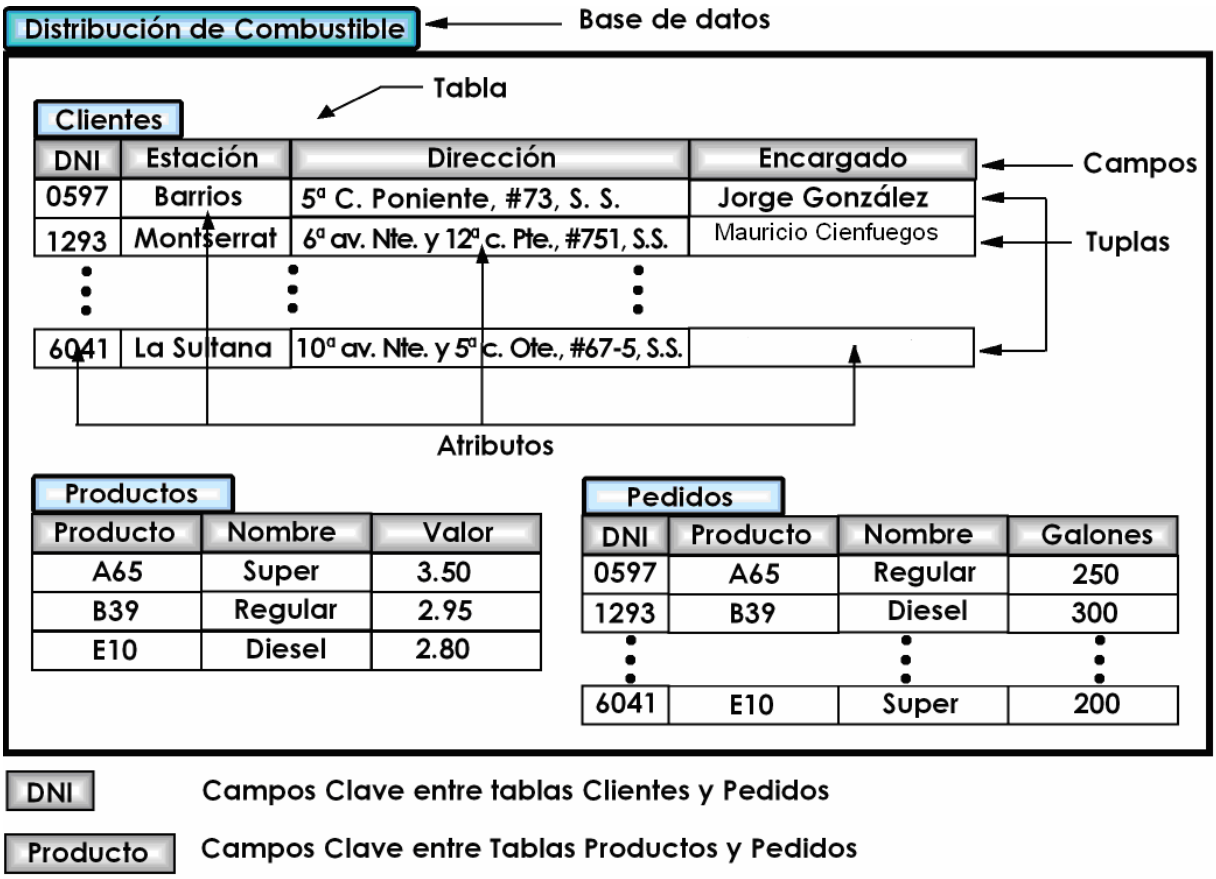


Figura 3.2.1 Ejemplo de una base de datos relacional.

Las tablas pertenecientes a una base de datos relacional pueden relacionarse entre sí utilizando campos clave comunes entre las tablas. Estas relaciones se utilizan para realizar búsquedas complejas y generar informes detallados, además de eliminar la necesidad de almacenar información redundante.

Una lista ordenada con los datos contenidos en un campo o grupos de campos, se le llama índice. Esta lista está diseñada para agilizar las búsquedas del motor de una base de datos.

Además de los datos explícitos de una base de datos, es decir los atributos, existe una información importante, llamada metadatos. Los metadatos se encuentran

recopilados en un conjunto de tablas del sistema, denominado catálogos. Los catálogos almacenan información acerca de las bases de datos, las tablas y la composición de dichas tablas, es decir, de toda la información presentada en la figura de ejemplo, exceptuando los atributos.

3.2.2 INTRODUCCIÓN A MICROSOFT ACCESS.

Access es una herramienta de desarrollo de base de datos, de la familia Microsoft Office disponible en un solo paquete, capaz de manejar bases de datos relacionales con total compatibilidad para compartir información con los restantes miembros de la familia de Microsoft Office, Microsoft Visual Basic y Microsoft SQL Server.

Con la introducción de Access 2000, es posible desarrollar y manipular bases de datos compatibles con SQL Server además de los archivos estándar de base de datos de Access. En esta versión de Access, es posible elegir entre dos plataformas nativas de base de datos: JET y MSDE (Microsoft Data Engine). Jet ha sido la plataforma estándar de las bases de datos de Access. MSDE es un motor de base de datos de Windows 95/98 compatible con código base de SQL Server 7.0. Esto significa que incluye todas las características de SQL Server 7.0 pero que puede ejecutarse en computadoras con Windows 95/98 en lugar de Windows NT.

Algunas de las características relevantes de Access son:

- Capacidad amigable para la recolección de datos, a través de la vista Hoja de datos, que es un mecanismo similar a una hoja de cálculo, o por medio de un formulario personalizado de apariencia semejante a la de todas las demás aplicaciones de Windows.
- Permite la creación de informes personalizados para la impresión o exportación de la información almacenada en la base de datos.
- En aplicaciones cliente servidor, ofrece una opción para el almacenamiento físico, pudiendo usar el archivo de base de datos de escritorio (del lado del cliente),

ofrecido por Access, o utilizar Microsoft SQL Server para almacenar datos del lado del servidor.

- Hace uso de índices para facilitar búsquedas y consultas.

3.2.2.1 OBJETOS EN UNA BASE DE DATOS CON ACCESS.

En Access el término base de datos se refiere al contenedor que aloja a la totalidad de los datos y sus objetos asociados.

Los siete objetos principales de Access son: tablas, consultas, formularios, informes, páginas macros y módulos. Aunque es posible que en otros programas de base de datos se llame base de datos al objeto en el que se alojan los datos, en Access ese objeto se llama tabla.

Access solo puede trabajar con una base de datos a la vez, pero en ella pueden hallarse una cantidad grande de objetos, como tablas, consultas y formularios. Todos ellos se almacenan en un archivo de Access.

- Tablas: Sirven para alojar los datos primarios de la base de datos. Las tablas están formadas por tuplas (registros) y campos.

- Consultas: Sirven para extraer sólo cierta información de una base de datos. Es capaz de seleccionar grupos de registros que cumplen ciertas condiciones. Los formularios pueden servirse de consultas para que en la pantalla aparezca sólo información específica. Los informes pueden servirse de consultas para imprimir únicamente ciertos registros. Las consultas pueden basarse en tablas u otras consultas. Pueden usarse para seleccionar, modificar, agregar o eliminar registros en una base de datos.

- Formularios: En general se utilizan para la introducción y presentación de datos. Las entradas de los formularios de datos permiten a los usuarios insertar datos en

las tablas de manera rápida, precisa y fácil. Los formularios presentan datos de un modo más estructurado que una tabla normal. Se permite llevar a cabo tareas como modificar, agregar, eliminar o visualizar registros de una tabla usando un formulario. Los formularios de presentación sirven para la exhibición selectiva de cierta información de determinada tabla.

- Los informes: Estos sirven para la presentación en formato impreso de datos seleccionados por el usuario. Pueden basarse en tablas, para mostrar todos los datos de una tabla, o en consultas, para mostrar únicamente la información que cumple con ciertos criterios. También pueden basarse en múltiples tablas y consultas para dar cuenta de complejas relaciones entre datos. Access dispone de numerosos informes predeterminados de fácil creación para la presentación de datos en cualquier modalidad requerida.

- Las páginas son una de las novedades de Access 2000, y su nombre formal es el de Páginas de acceso a datos (Data Access Pages). Son documentos HTML que pueden relacionarse directamente con datos de una base de datos. Estos documentos son muy similares a los formularios de Access, pero están diseñadas para ser vistos con Internet Explorer. Una de las principales diferencias entre las Páginas de acceso a datos y los formularios es que las primeras se guardan en un archivo distinto al de la base de datos. Esto se debe a que las páginas están diseñadas para uso con un explorador de Internet y en ellas se utiliza HTML dinámico.

- Macros: Contribuyen a automatizar tareas repetitivas sin tener que escribir código complejo o aprender un lenguaje de programación. Los macros son simplemente un conjunto de acciones, cada una de las cuales desempeñan una tarea específica en un proyecto de Access.

- Módulos: son conjuntos de procedimientos de Visual Basic para aplicaciones (VBA). Access se sirve del mismo lenguaje de programación integrado que las demás aplicaciones de Microsoft Office. El desarrollo en VBA es esencialmente

idéntico al que se realza con la herramienta de desarrollo de aplicaciones Microsoft Visual Basic. Como lenguaje, VBA es un subconjunto del lenguaje estándar Visual Basic. En Access permite crear funciones y procedimientos personalizados, así como controlar programáticamente el motor de base de datos subyacente de Access.

3.2.2.2 RELACIONES.

Una de las herramientas más importantes de las bases de datos son las relaciones. Las tablas deben relacionarse entre sí para que la información de una de ellas pueda ser accesada por otras. En la mayoría de los casos varias tablas se relacionarán entre sí. Tal relación se establece mediante la presencia de ciertos campos que comparten valores comunes con otras tablas. No es necesario para ello que los nombres de los campos sean iguales, pero los valores tienen que coincidir.

La aplicación de un correcto diseño de tablas y relaciones permite evitar el almacenamiento de los mismos datos en dos lugares distintos. La eliminación de datos duplicados no sólo ahorra tiempo, sino que contribuye a preservar la exactitud de los datos.

CAPÍTULO IV: *DISEÑO DEL HARDWARE*

4.1. PANORAMA GENERAL ENTORNO A LA ARQUITECTURA DE LA APLICACIÓN.

La arquitectura propuesta para la aplicación se presenta en la figura 4.1, en ella se observa una separación entre el equipo móvil y la estación central, así como los circuitos o dispositivos contenidos en cada una de estas ubicaciones.

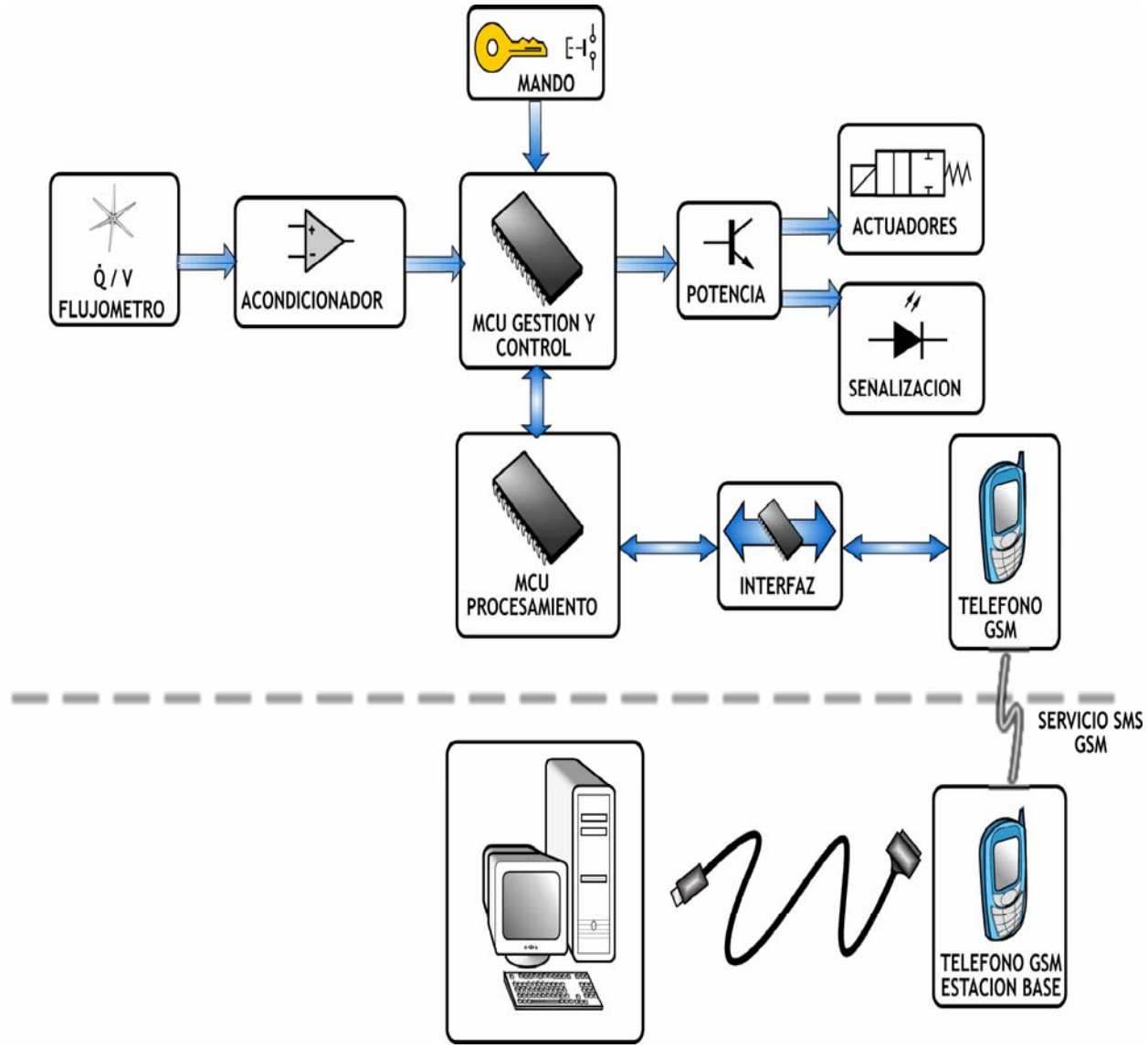


Figura 4.1 Arquitectura de la aplicación.

La descripción que en éste capítulo se realiza, está basada en la estrecha relación entre algunas etapas, logrando de esta forma cinco bloques principales: Bloque de instrumentación, Bloque de Control, Bloque de Telecomunicación, Interfaz de comunicación y Bloque de la estación central. En cada uno de estos bloques se describen sus funciones primarias, las funciones a desempeñar por parte de cada circuito o dispositivo interno y las consideraciones para el diseño y/o elección.

4.2. BLOQUE DE INSTRUMENTACIÓN: FLUJOMETRO Y SU ACONDICIONADOR DE SEÑAL.

Tanto las características físicas de los combustibles, como la volatilidad y la rápida combustión, así como aspectos técnicos relacionados al proceso de descarga y medición volumétrica de los mismos; conllevan a que los dispositivos que interactúan con dichos materiales sean costosos y de limitada disponibilidad en el mercado, si no se es una empresa del sector.

Por lo que, para materializar el prototipo, se utilizará el sistema de medición didáctico basado en un caudalimetro de aspa giratoria de la marca Lucas-Nülle®, propiedad del Centro de Investigación y Transferencia de Tecnología (CITT), junto con tuberías y electroválvulas que poseen la capacidad de transportar pequeñas cantidades de fluido.

Con el objetivo de no dejar corta la propuesta del diseño, se presentan apartados relacionados a dispositivos de la industria de hidrocarburos en aquellas etapas que se consideran pertinentes, tales como la fase de adquisición y acondicionamiento de la señal medida.

4.2.1. SENSOR DE FLUJO UTILIZADO PARA EL DISEÑO DEL SISTEMA.

Debido a las propiedades químicas y físicas de los hidrocarburos líquidos, estos son catalogados como líquidos no conductivos e inflamables, los flujometros que por

el modo de operación se ajustan a éstas peculiaridades, son los tipos de caudalimetros con partes móviles, ya sea de desplazamiento positivo o los de turbina, y los flujómetros no invasivos, como los ultrasónicos.

4.2.1.1. SENSOR DE FLUJO PARA EL PROTOTIPO.

El sistema de medición de caudal de Lucas Nülle utiliza un flujómetro de chorro único de aspas giratorias o turbina, con una capacidad de medición mínima de 0.04 lts./min. hasta una máxima de 6 lts./min. con una tolerancia a plena escala de ± 1%.

Las aspas disponen de barras de ferrita, que junto a un detector inductivo y un circuito de pre acondicionamiento, registran el movimiento angular de las barras, produciendo una señal pulsante que es proporcional a la cantidad de líquido que fluye por el sensor.

Como el sistema de medición forma parte de un módulo de aprendizaje de control automático analógico, éste entrega a la salida una señal de corriente directa proporcional a la cantidad de volumen por unidad de tiempo, de acuerdo a la siguiente relación:

$$lV \cong 0.1\frac{l}{m} \qquad \text{Ecuación 4.1}$$

Finalmente, el medidor de flujo como bloque, es decir con la alimentación del sistema didáctico y la etapa de acondicionamiento, entrega una señal máxima de salida de +15V, por lo que el flujo volumétrico máximo medible se limita a 1.5 lts./min.

4.2.1.2. PROPUESTA DE FLUJOMETRO PARA UN SISTEMA INDUSTRIAL REAL.

Tal como se describe al inicio de éste apartado, los flujometros que por sus características son aplicables a la medición de flujo de combustibles líquidos son los flujometros ultrasónicos y los de partes móviles, siendo éste último el elegido para presentarlo como propuesta para una aplicación industrial.

Una de las razones por la que se elige un flujómetro de aspas sobre un ultrasónico es debido a consideraciones mecánicas de las instalaciones y disposición de tuberías. Para un flujómetro ultrasónico se requieren distancias mas largas para la ubicación del sistema de medición con respecto a los puntos de cambio de dirección del flujo (codos). Éste requisito se debe cumplir con el objetivo de reducir la turbulencia en el proceso de medición, requerimiento un poco difícil de satisfacer en los equipos de transporte de combustibles.

El sensor propuesto es un sensor de la marca SeaMetrics de una sola parte móvil, formado por un rotor helicoidal de Kynar sobre un eje de Zirconio cerámico, cuyo movimiento angular es detectado y procesado electrónicamente a través de módulos intercambiables que mejor se ajusten a las necesidades de medición.

4.2.1.2.1. CARACTERÍSTICAS MECÁNICAS.

SeaMetrics ofrece las características mecánicas enlistadas en la tabla 4.1 para el grupo de medidores de la familia WT-S. En cuanto a la instalación de las tuberías, SeaMetrics recomienda una distancia mínima de cinco veces el diámetro de la tubería del flujo ascendente y una distancia de tres veces el diámetro del lado descendente del flujo. La figura 4.2 muestra las condiciones mínimas para la instalación del flujómetro.

Presión Máxima	200 psi (14 bar)
Temperatura Máxima	200 ^{o}F (93 ^{o}C)
Precisión	± 1% FS
Rango mínimo de flujo (4”)	6 GPM
Rango máximo de flujo (4”)	600 GPM
Rango mínimo de flujo (6”)	12 GPM
Rango máximo de flujo (6”)	1200 GPM

Tabla 4.1 Resumen de características mecánicas de los flujometros WT-S. [28]

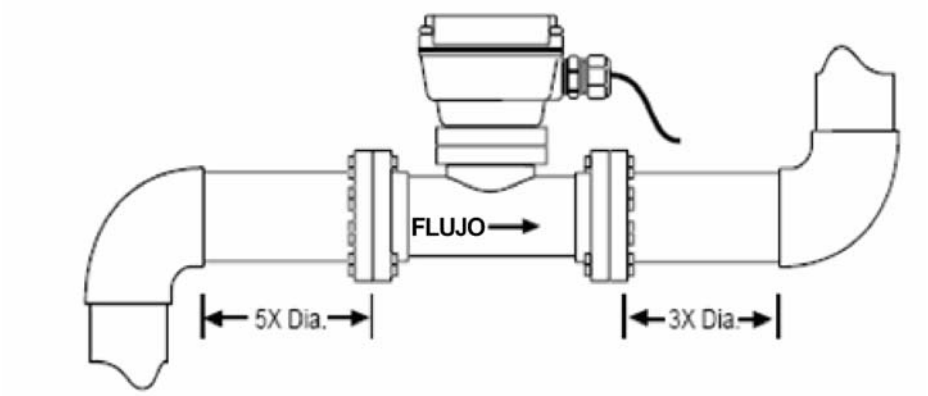


Figura 4.2 Requerimientos de instalación para el flujómetro WT-S. [28]

4.2.1.2.2. CARACTERÍSTICAS ELÉCTRICAS.

Las salidas eléctricas del sensor pueden ser obtenidas utilizando cualquiera de los cuatro módulos ofrecidos por SeaMetrics, entre ellos se opta por el módulo WT102 que ofrece salida analógica de 4mA a 20mA.

Esta opción se considera oportuna ya que por hardware se puede manipular hasta obtener una señal con las mismas características que el sensor didáctico, con lo que se lograría cambios mínimos en la programación del bloque de control, además de ofrecer un diagnostico rápido si existe ausencia de señal

eléctrica en las líneas de medición, ya que esto justificaría un circuito abierto en el lazo del sensor y no la medición de un flujo nulo.

El fabricante ofrece las siguientes características eléctricas para el módulo analógico WT102:

Rango de salida	4 - 20 mA
Lazo de alimentación	12 – 36 VDC (Aislada)
Precisión	± 1%
Tiempo de respuesta	3 seg., 95% FS
Cable	#22 AWG, 3 conductores, 18 ft. (6m)

Tabla 4.2 Características eléctricas del módulo WT102. [28]

4.2.2. EL CIRCUITO DE ACONDICIONAMIENTO.

En el proyecto, el circuito de acondicionamiento se encarga de proporcionar una señal eléctrica compatible con las características requeridas por el circuito de control, con lo que básicamente sirve de pasarela entre la salida de la etapa de adquisición y la entrada de los canales de conversión análogo – digital del sistema de control basado en microcontroladores de gama media de la familia PIC.

De igual forma que en la etapa de adquisición, se presentan dos circuitos de acondicionamiento, uno, utilizado para el prototipo, y el segundo para un posible diseño real aplicado a la industria de hidrocarburos. Este último estaría ligado al sensor de flujo WT-S de SeaMetrics presentado anteriormente.

4.2.2.1. CIRCUITO DE ACONDICIONAMIENTO PARA EL PROTOTIPO.

Debido a que el prototipo está basado en el flujómetro didáctico de Lucas Nülle que entrega un voltaje de salida en el rango de 0V a +15V, el circuito de

acondicionamiento deberá entregar, a petición del bloque de control, una señal de DC con una amplitud máxima de +5V a los canales de conversión análogo – digital a una impedancia máxima de salida de 2.5 kΩ.

En base a las características anteriores, la solución se presta para un circuito sencillo de un solo amplificador operacional en configuración diferencial, con una función de transferencia menor que la unidad.

Durante la fase de simulación no presentó problema alguno al ser alimentado con una sola fuente, pero durante las pruebas con el sensor, el circuito generó una señal de salida con valores no esperados, por lo que se tuvo que adicionar una fuente para la alimentación negativa del circuito.

4.2.2.1.1. DISEÑO DEL CIRCUITO DE ACONDICIONAMIENTO.

Los parámetros eléctricos considerados para el diseño se enlistan en la tabla siguiente:

Parámetro Eléctrico	**Magnitud**	**Circuito fuente / Circuito destinatario**
Rango voltaje de entrada	0V – 15V DC	Salida flujómetro Lucas Nülle
Rango voltaje de salida	0V – 5V DC	Canales ADC del Microcontrolador
Impedancia de salida (Máx.)	2.5 kΩ	Canales ADC del Microcontrolador
Alimentación	± 9V	El sistema dispone de una fuente de +9V y +5V alimentadas por una sola fuente unipolar de +12V, por lo que para obtener –9V será necesario utilizar una fuente externa adicional.

Tabla 4.3 Parámetros eléctricos del circuito de acondicionamiento del prototipo.

Del concepto básico de función de transferencia y de su concordancia con el montaje diferencial, se obtienen las siguientes ecuaciones respectivamente:

$$H = \frac{v_o}{v_i}$$ Ecuación 4.2

$$H = \frac{R_f}{R_i}$$ Ecuación 4.3

Igualando estas dos ecuaciones junto con los datos de la tabla 4.3, se obtiene la siguiente relación:

$$\therefore \frac{R_f}{R_i} = \frac{1}{3}$$ Ecuación 4.4

Por otra parte, del teorema de Miller se tiene que:

$$Z_o = \left(\frac{A_{vol}}{1 + A_{vol}} \right) \bullet Z_f$$ Ecuación 4.5

Si se considera que para el amplificador operacional LM741 la ganancia de lazo abierto, Avol = 200,000 y que la $Z_{Omax.}$ del circuito debe ser de 2.5kΩ, entonces, R_f estará limitado por el siguiente valor:

$$Z_f \cong 2.5k\Omega$$

$$\therefore R_f \leq 2.5k\Omega$$ Ecuación 4.6

Por último, relacionando las ecuaciones 4.4 y 4.6 y asumiendo un valor para R_f, se obtiene que:

$$\text{Si } R_f = 1.2k\Omega$$

$$R_i = 3.6k\Omega$$

En la figura 4.3 se presenta el circuito acondicionador para un canal de conversión análogo – digital.

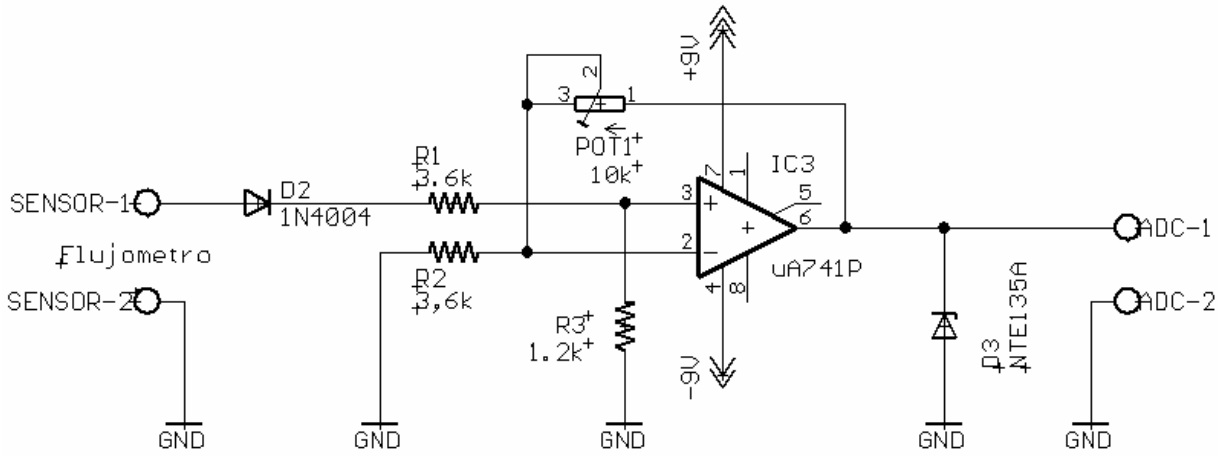


Figura 4.3 Circuito acondicionador para un canal ADC.

Es de notar que el montaje es un amplificador diferencial, que realiza la diferencia de voltajes con respecto a tierra, del potencial proveniente del sensor de flujo y de tierra misma. En la salida del circuito se encuentra un diodo Zener de 5.1V que realiza la función de limitación de voltaje con el objetivo de no dañar el canal ADC del microcontrolador, sí se llegase a presentar un sobrevoltaje en la entrada del circuito.

También está presente un diodo Rectificador que protege al circuito de conexiones cruzadas en la entrada del mismo. El efecto colateral de este diodo es disminuir el voltaje proveniente del sensor, por lo que se utiliza un potenciómetro de 10kΩ como elemento de retroalimentación para compensar el voltaje de entrada, el valor del potenciómetro debería rondar el valor asumido anteriormente, es decir el de 1.2kΩ.

4.2.2.2. CIRCUITO DE ACONDICIONAMIENTO PARA UN SISTEMA INDUSTRIAL REAL.

Como el módulo de salida del flujómetro WT-S de SeaMetrics proporciona una salida de corriente directa en el rango de 4mA a 20mA, será necesario que el circuito de acondicionamiento realice la conversión de corriente a voltaje, con valores que se ubiquen dentro del rango descrito anteriormente para el circuito de acondicionamiento del prototipo.

Para realizar dicha tarea, se utilizará una resistencia de muestreo que se encargará de reflejar el potencial eléctrico producido por la corriente de salida del flujómetro, por otro lado un circuito divisor de tensión servirá para la corrección del offset producido cuando el sensor entregue a la salida una corriente de 4mA.

Ambos potenciales, es decir, el producido en la resistencia de muestreo y en el circuito divisor de tensión, se introducirán en un circuito diferencial basado en un amplificador operacional OP07, cuyo objetivo será entregar un voltaje entre 0V y 5V a una impedancia de salida menor o igual que 2.5kΩ.

Se escoge el OP07 por ser un dispositivo que ofrece altas prestaciones para circuitos de instrumentación, tales como bajo offset, alto CMRR, alta impedancia de entrada y estabilidad ante cambios bruscos de temperatura.

4.2.2.2.1. DISEÑO DEL CIRCUITO DE ACONDICIONAMIENTO.

El resumen de las características ligadas al circuito de acondicionamiento se enlista en la tabla siguiente:

Parámetro Eléctrico	Magnitud	Circuito fuente / Circuito destinatario
Rango corriente de entrada	4mA – 20mA	Salida flujómetro WT-S SeaMetrics
Rango voltaje de salida	0V – 5V DC	Canales ADC del Microcontrolador
Impedancia de salida (Máx.)	2.5 kΩ	Canales ADC del Microcontrolador
Alimentación	+ 12V	El circuito debe alimentarse de la batería del equipo de transporte, esta alimentación debe estar protegida para evitar la introducción de EMIs.

Tabla 4.4 Parámetros eléctricos del circuito de acondicionamiento para un diseño industrial.

En la figura 4.4 se presenta el diseño genérico propuesto para el acondicionamiento de la señal proveniente del sensor. Al igual que el circuito para el prototipo, éste circuito rige su comportamiento bajo las ecuaciones 4.2 y 4.3, así como la restricción de cumplir la condición definida por la ecuación 4.6.

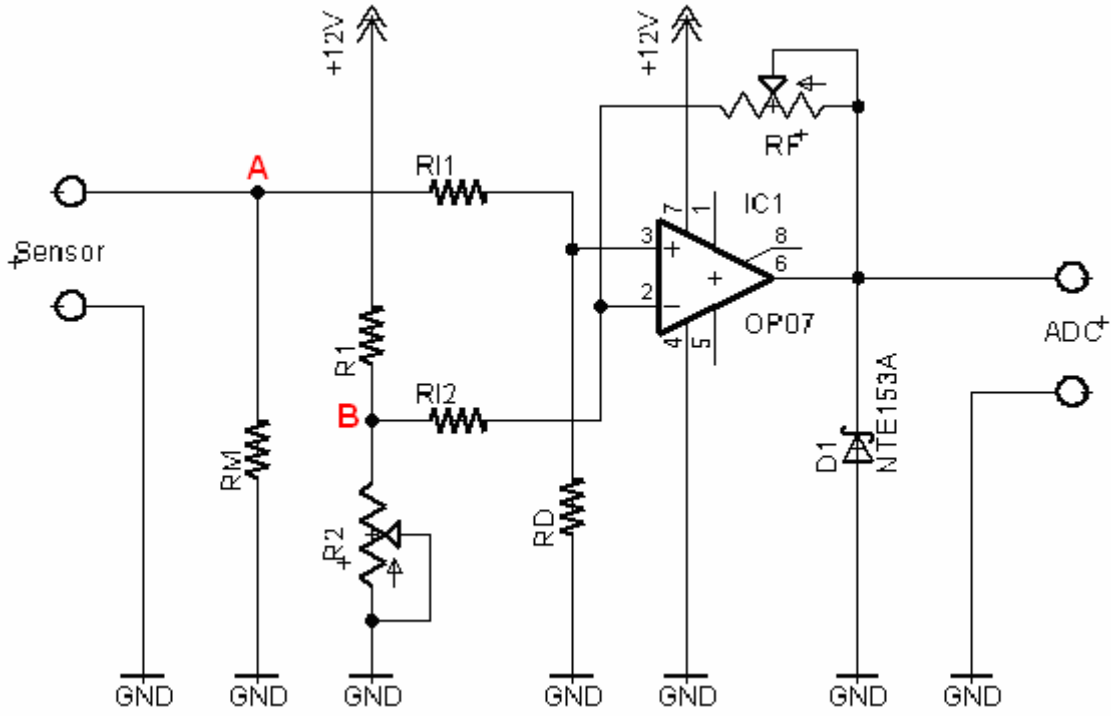


Figura 4.4 Propuesta de diseño para el circuito acondicionador.

Ya que el flujómetro devuelve una corriente proporcional entre 4mA y 20mA se escoge una resistencia de muestreo (R_M) de 560Ω, de tal forma que, bajo una primera aproximación, el voltaje reflejado sobre ella se encuentre en el rango de 2.24V y 11.2V, evitando con ello alguna saturación en la respuesta del amplificador operacional.

Considerando que en el nodo A, se producirá un divisor de corriente, es necesario asumir un valor cercano al límite para R_D (para el montaje diferencial $R_F = R_D$) con el objetivo de que la rama formada por R_{i1} y R_D, consuman la menor cantidad de corriente e interfieran sobre el voltaje que se obtendrá en el nodo antes mencionado.

Siempre bajo la aproximación de que toda la corriente del sensor atraviesa R_M, el máximo valor de voltaje entrada será +11.2V, esperando obtener una salida con magnitud de +5V. Ocupando la ecuación 4.2, con la salvedad que V_i será el voltaje diferencial entre V_+ y V_-, es decir +8.96V, la función de transferencia tendrá el valor adimensional de 0.558. Con éste dato, asumiendo un valor ohmico de 2.4kΩ para R_F y sustituyéndolos en la ecuación 4.3, se obtendría una R_i ($R_{i1} = R_{i2}$) con valor de 4.3kΩ.

Igualmente bajo la misma aproximación, el voltaje mínimo en los terminales de la misma sería de 2.24V, por lo que el circuito divisor de tensión tendrá que ajustarse para igualar dicho voltaje y de esta forma obtener 0V a la salida, que equivaldría a 4mA de corriente de entrada y obviamente al menor flujo perceptible por el sensor. Para ello se utiliza una resistencia fija y una variable sumando unos cuantos miles de ohmios y evitando así disipaciones excesivas de potencia.

Con los valores asumidos y calculados anteriormente se obtendrán ciertas desviaciones en el rango de voltajes esperados en el nodo A y en la salida del circuito acondicionador, por lo tanto, para efectuar las correcciones, R_F y R_2

serán potenciómetros de precisión, logrando adicionalmente, cierto margen de maniobra al momento de calibrar el sistema completo con el sensor.

Al efectuar la simulación del circuito anterior en PSpice Student™ se obtienen los resultados esperados con valores satisfactorios, realizando ciertos ajustes a los valores de resistencias, que se muestran en la siguiente tabla:

Relaciones obtenidas entre corrientes de entrada y voltajes de salida					
$I_{i\ FALLO}$	$V_{o\ FALLO}$	$I_{i\ min}$	$V_{o\ min}$	$I_{i\ max}$	$V_{o\ max}$
0 mA	0 V	4 mA	0.03 V	20 mA	5.001V

Valores óhmicos ajustados			
R_M:	560 Ω	R_i ($R_{i1} = R_{i2}$):	3.9 kΩ
R_1:	1 kΩ	R_D:	2.4 kΩ
R_2:	212.5Ω	R_F:	2.43 kΩ

Tabla 4.5 Datos obtenidos de la simulación para el circuito de acondicionamiento.

4.3. BLOQUE DE CONTROL: MICROCONTROLADORES, DISPOSITIVOS DE MANDO Y SEÑALIZACIÓN, Y ACTUADORES.

El centro del bloque de control está formado por dos microcontroladores PIC16F877A que se encargan del procesamiento de la señal analógica, auxiliándose del circuito de acondicionamiento, también interactúa con el usuario a través de una serie de dispositivos de mando y señalización e interviene sobre los actuadores apoyándose de una interfaz de potencia basada en el circuito integrado ULN2803, de acuerdo al algoritmo de funcionamiento establecido.

En último lugar espera, gestiona y mantiene comunicación con el teléfono móvil a través de un circuito de interfaz serie, logrando con ello la activación del propio sistema electrónico, de cara al usuario, al inicio de una transacción y la transmisión de los volúmenes descargados al final de la misma.

4.3.1. EL MICROCONTROLADOR PIC 16F877A.

Dentro del diseño de la aplicación, el microcontrolador debe realizar las siguientes funciones:

- Esperar comunicación tipo serie con el dispositivo móvil para activar el sistema.
- Operar a voluntad del usuario, de acuerdo al protocolo de funcionamiento establecido.
- Obtener las conversiones de la señal analógica acondicionada y procesar dichos valores digitalizados para obtener información útil.
- Gestionar el envío de la información, vía comunicación serie, de la transacción realizada hacia el teléfono móvil, para su posterior procesamiento.

Para cumplir con dichas premisas es necesario que el microcontrolador a escoger posea al menos tres canal de conversión A/D, puertos I/O digitales, líneas para comunicación serie asíncrona, temporizadores, entre otros recursos. Los microcontroladores propuesto son el MCPHC12 de Motorola y el PIC16F877A de Microchip, siendo este último el elegido tanto por sus características técnicas como su bajo costo.

4.3.1.1. ARQUITECTURA INTERNA.

El PIC 16F877A es un microcontrolador de gama media de la compañía Microchip, que en un encapsulado PDIP de cuarenta pines, ofrece de forma multiplexada y a través de registros de control, cinco puertos para entradas salidas digitales, un canal de comunicación serie (MSSP/USART), ocho canales de conversión análogo digital, dos canales de comparación analógica, dos módulos de captura y comparación de eventos discretos y tres temporizadores programables.

Desde el punto de vista del núcleo, este microcontrolador posee un procesador RISC con un set de 35 instrucciones, que opera a una frecuencia máxima de 20 MHz, dispone de 8K (de 14 bits) de memoria Flash para el código del programa, 368 bytes de RAM para el almacenamiento general de datos y 256 bytes de EEPROM para datos de usuario.

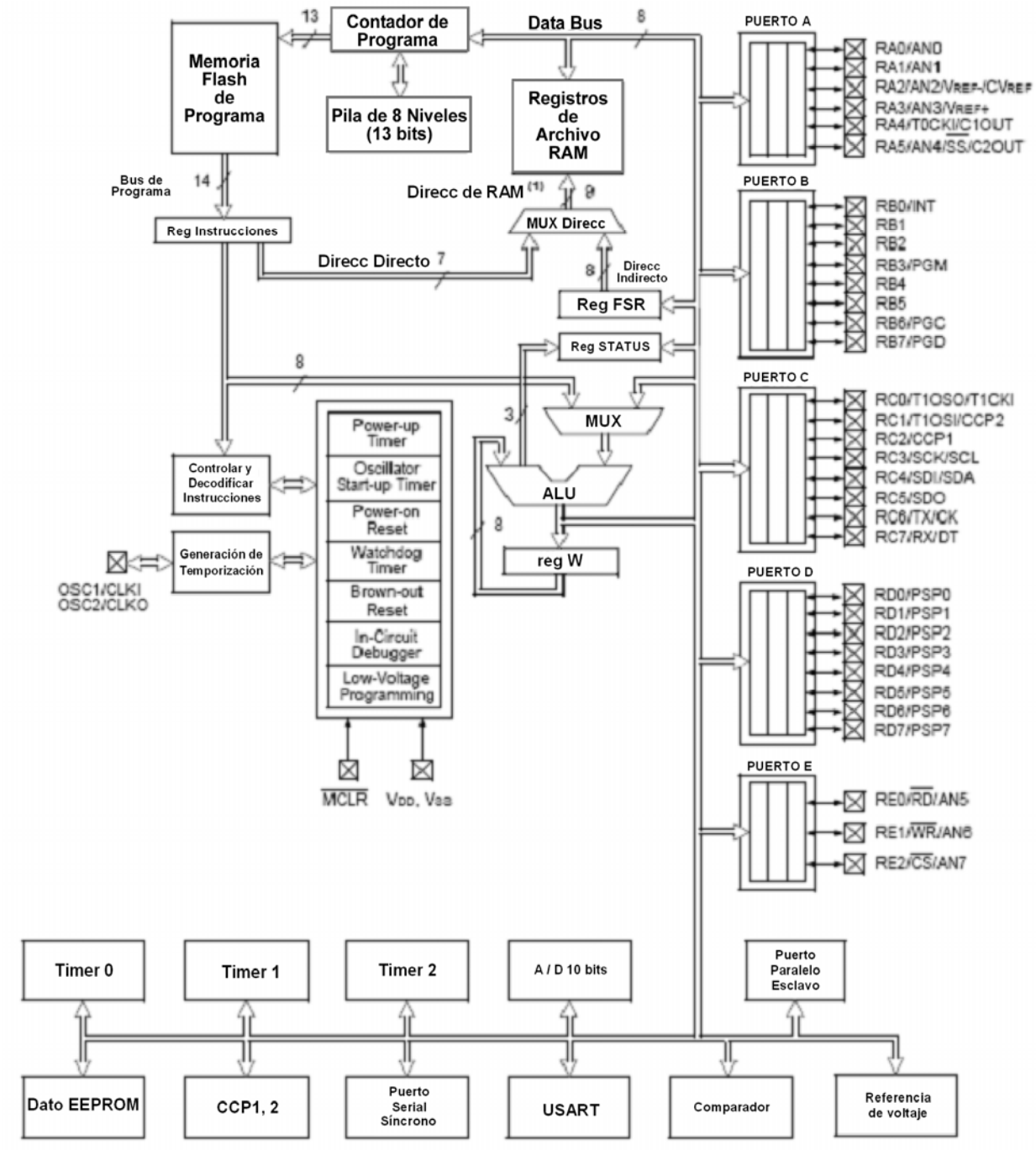


Figura 4.5 Diagrama de bloques del PIC16F877A. [29]

La figura 4.5 ilustra una idea de las interacciones entre los distintos subsistemas, de igual forma presenta el modelo de programación, es decir la estructura de la manipulación de los datos que soporta el conjunto de instrucciones.

4.3.1.2. MAPA DE MEMORIA.

Los microcontroladores PIC utilizan arquitectura Harvard, la cual permite la coexistencia de una memoria de programa y una de datos con su respectivos buses, permitiendo al procesador acceder simultáneamente a ambos bloques de memoria.

Cabe destacar, que la memoria de datos esta dividida entre la RAM de propósito general y los registros de funciones especiales (SFR). Los SFR son utilizados para controlar los periféricos.

4.3.1.2.1. MAPA DE LA MEMORIA DE PROGRAMA (FLASH)

Los microcontroladores PIC de gama media poseen un contador de programa de 13 bits con capacidad de direccionamiento de un espacio de memoria para programa de 8K x 14 bits, logrando con esto instrucciones de una sola “palabra” y reservar de esta forma dicho espacio de memoria únicamente para almacenar el programa a ejecutar.

Esta porción de memoria esta dividida en cuatro páginas de 2K cada una (0h -7FFh, 800h – FFFh, 1000h – 17FFh, y 1800h – 1FFFh), únicamente se reserva la dirección 0000h y la 0004h para los vectores de Reset y de Interrupción, respectivamente. La figura 4.6 presenta el mapa de memoria junto con la relación entre la pila de ocho niveles y el contador de programa.

Para saltar entre las páginas de la memoria de programas, los bits mas significativos del contador de programa (PC) deben ser modificados, a través

de la escritura del valor deseado en un registro de funciones especiales llamado PCLATH (Program Counter Latch High). Si la secuencia de instrucciones esta en ejecución, este proceso se realiza sin intervención del usuario.

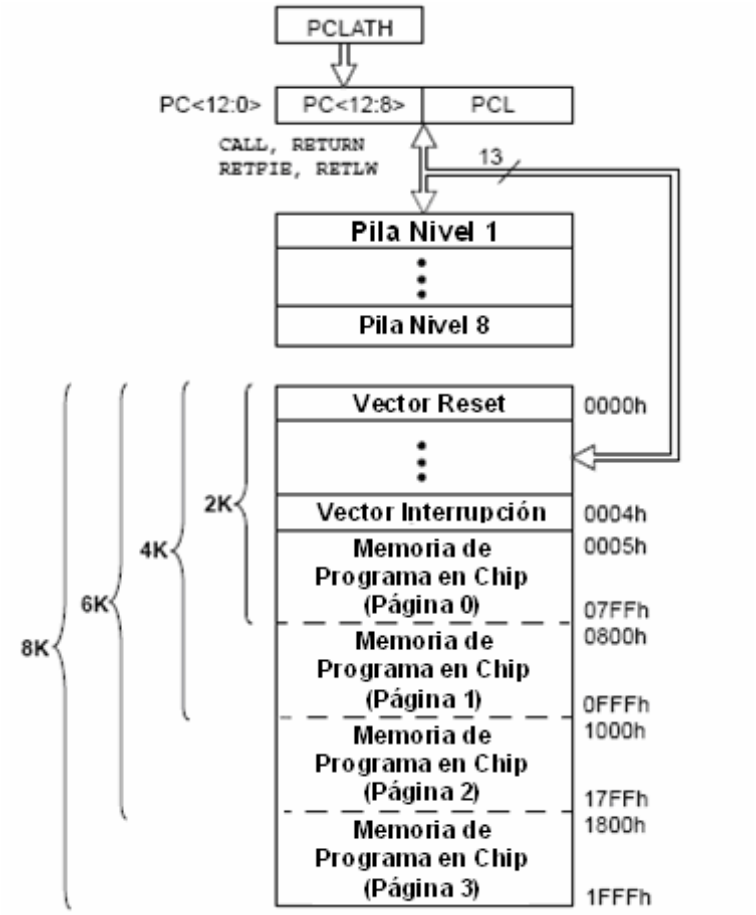


Figura 4.6 Organización de la memoria de programa del PIC16F877A. [30]

4.3.1.2.2. MAPA DE LA MEMORIA DE DATOS (RAM).

La memoria de datos esta particionada en cuatro bancos, los cuales contienen los Registros de Propósitos Generales (GPR) y los Registros de Funciones Especiales (SFR). Cada banco esta formado por 128 bytes, formando en total 512 bytes de RAM, pero únicamente 368 bytes repartidos en los cuatro bancos están disponibles para el usuario. La figura 4.7 presenta la distribución de registros en los cuatro bancos.

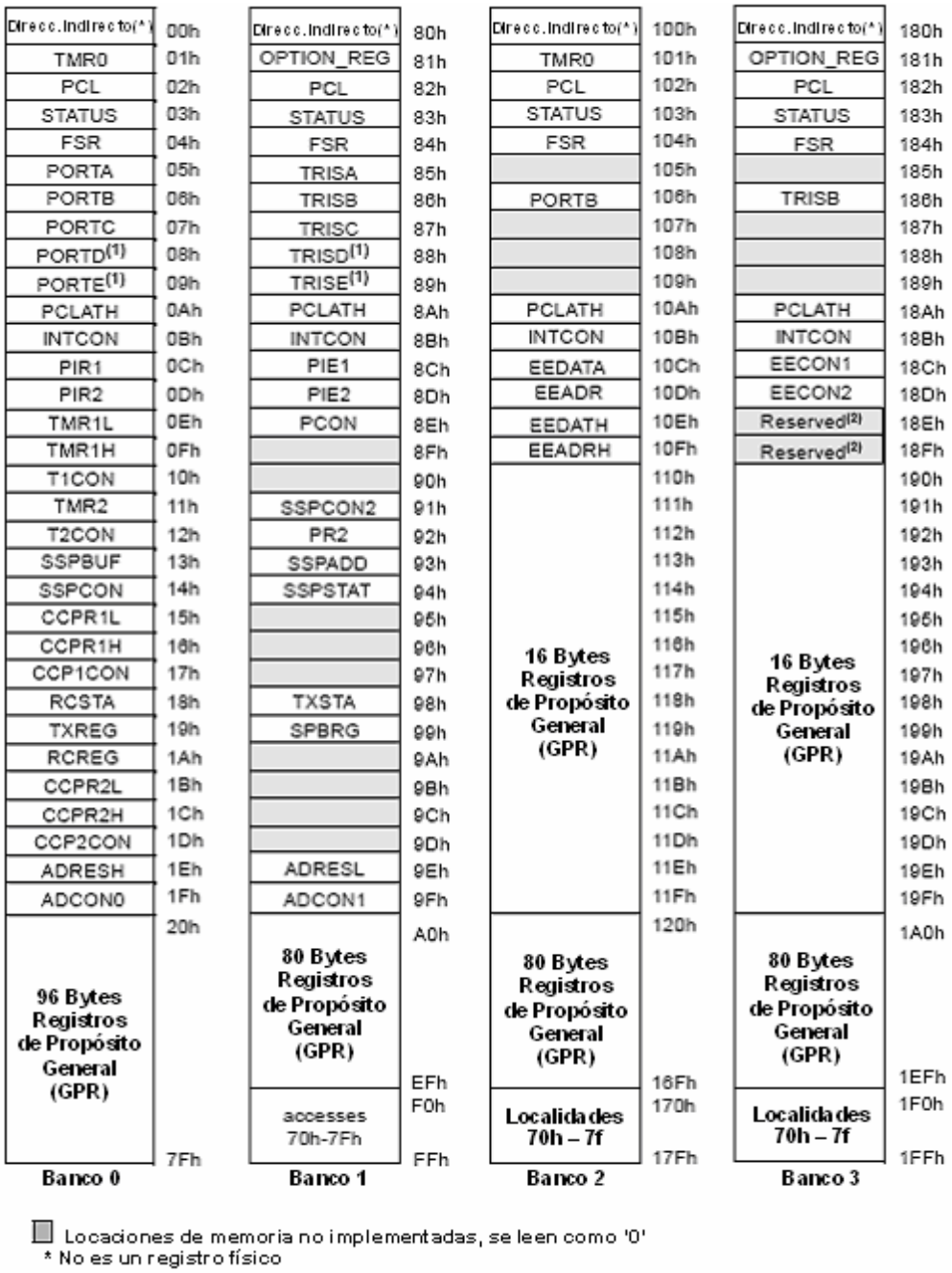


Figura 4.7 Organización de la RAM. [29]

Las localidades bajas de cada banco están reservadas para los SFR. Estos registros se pueden clasificar en dos tipos: los registros de control para el núcleo (CPU) y los encargados de manipular el modo de operación de los periféricos del microcontrolador.

Los GPR, se encuentran en las localidades altas de cada banco, pudiendo acceder a ellos, mediante un direccionamiento directo o indirecto, que se escoge a través de la manipulación del Registro Especial FSR.

4.3.1.3. SUBSITEMA DE ENTRADAS - SALIDAS DIGITALES.

Las entradas-salidas digitales son llamadas pines de propósito general, que en la mayoría de los casos, estos pines físicamente se comparte con los subsistemas periféricos especiales. En general, cuando un periférico no esta en funcionamiento, los pines asociados son usados como líneas de I/O digitales.

La tabla 4.6 presenta un resumen de los cinco puertos disponibles en el microcontrolador, detallando el o los periféricos asociados a cada puerto, así como los registros involucrados para la gestión operativa de los mismos.

Puerto	**I/O Digitales**	**Periférico asociado (multiplexado)**	**Registros de control.**
A	6 bits	ADC ó Comparador analógico	PORTA, TRISA CMCON, CVRCON, ADCCON1
B	8 bits	Líneas para la programación del dispositivo.	PORTB, TRISB
C	8 bits	Puerto de comunicación (USART/ I^2C), salidas PWM, Comparador de eventos discretos.	PORTC, TRISC
D	8 bits	Puerto paralelo esclavo de 8 bits	PORTD, TRISD
E	3 bits	ADC ó Bits de estado/control para el puerto paralelo.	PORTE, TRISE, ADCCON1

Tabla 4.6 Resumen de puertos en el microcontrolador PIC16F877A. [29]

Los registros de control enlistados anteriormente, son los encargados de manipular el modo de operación y la mantener la información a ser escrita o leída desde el núcleo del microcontrolador. Cuando un puerto opera únicamente como puerto de propósito general, los dos únicos registros asociados a estos, son TRIS<x> y PORT<x>.

El registro TRIS es el registro encargado de manipular la dirección del flujo de datos de cada bit en el puerto especificado, mientras que el registro PORT es la localidad de memoria donde se almacena (latch) la información binaria a ser escrita desde el procesador hacia los pines del puerto. Cuando es necesaria la lectura de los niveles lógicos presentes en los pines, ésta se almacena siempre en el registro PORT directamente desde los pines del dispositivo (non-latch).

Algunos detalles del párrafo anterior, así como la lógica de control para los procesos de lectura/escritura se presentan en la figura siguiente:

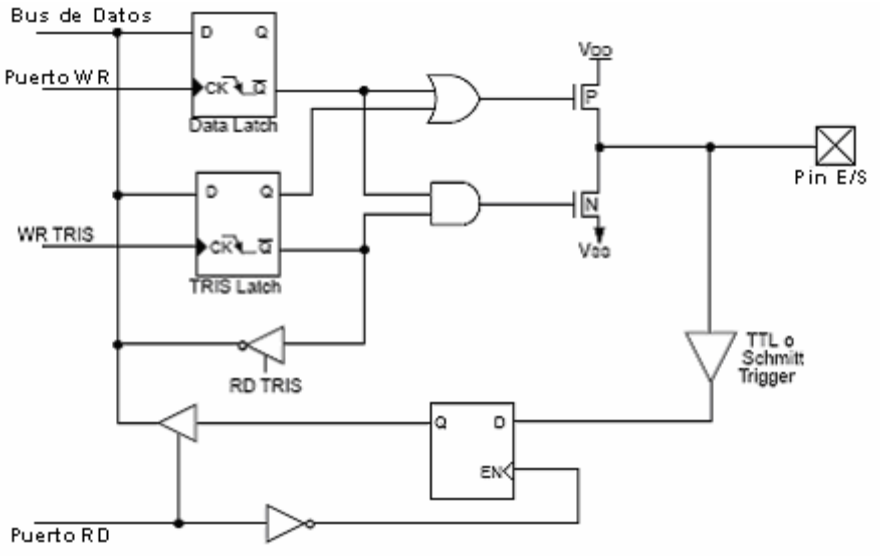


Figura 4.8 Diagrama interno típico de un puerto de propósito general. [30]

4.3.1.4. SUBSITEMA DE CONVERSION ANALÓGICO - DIGITAL.

El subsistema ADC del microcontrolador, dispone de ocho canales de conversión distribuidos en los puertos A y E (ver figura 4.4), con una resolución de 10 bits por canal. Para la configuración y operación del periférico, se utilizan los registros mostrados en la tabla 4.7, destacando que ADCON0 controla la operación del modulo A/D y ADCON1 configura las funciones de cada pin del puerto.

Nombre	Descripción	Dirección
ADRESH	Parte alta resultante de la conversión A/D (2 bits)	1Eh
ADRESL	Parte baja resultante de la conversión A/D (8 bits)	9Eh
ADCON0	Registro de control 0 del subsistema A/D	1Fh
ADCON1	Registro de control 1 del subsistema A/D	9Fh

Tabla 4.7 Resumen de registros asociados al subsistema de conversión A/D.

La figura 4.9 muestra la organización del modulo de conversión A/D, de la cual se identifican los bits CHS2..CHS0 del registro ADCON0, se utilizan para seleccionar el canal de conversión que será procesado; por otro lado los bits PCFG3..PCFG0 contenidos en el registro ADCON1, configuran el voltaje de referencia del conversor, ya sea tomándolo de la alimentación del microcontrolador o de alguna fuente externa.

La secuencia de conversión se ilustra en la figura 4.10, en donde se establece un tiempo de muestreo conformado a su vez por el tiempo de adquisición, que dependerá de la interacción eléctrica de los elementos del circuito de entrada de cada canal, y del tiempo necesario para realizar la conversión misma que se estima en unas $12T_{AD}$, siendo el valor de T_{AD} seleccionable por software a través del registro ADCON0.

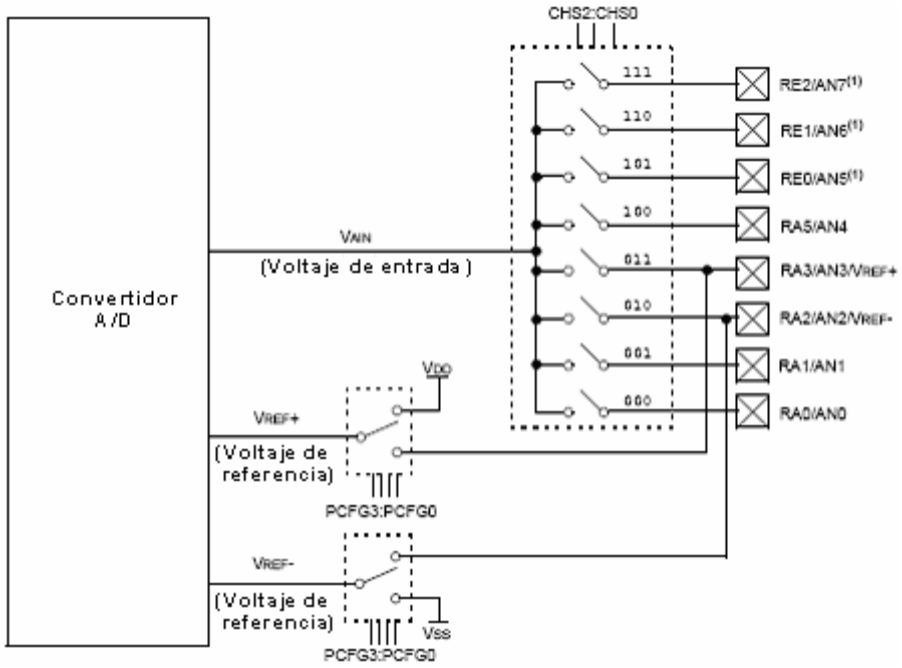


Figura 4.9 Diagrama de bloques del conversor A/D. [29]

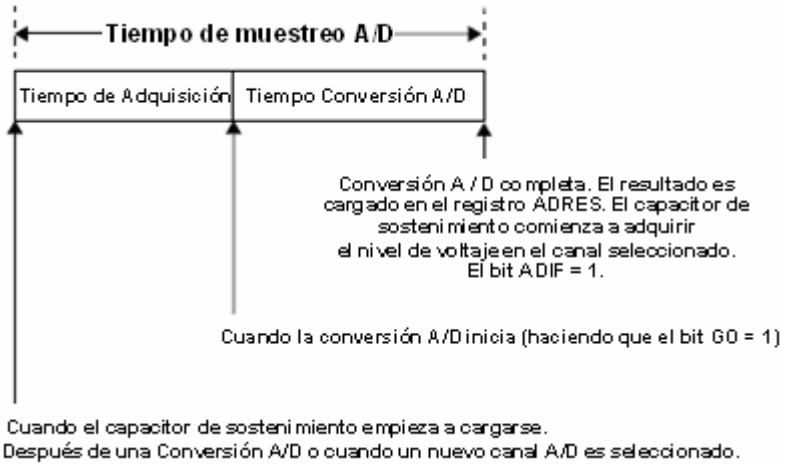


Figura 4. 10 *Secuencia de conversión A/D [30]*

En la entrada de cada canal de conversión, se dispone de un circuito de muestreo y un capacitor de carga $C_{HOLD,}$ encargados de estabilizar el voltaje de entrada, a fin de evitar cambios abruptos hacia el circuito conversor, ver figura 4.11. En esta figura también se hace evidente la presencia de la impedancia de la fuente de excitación, R_S, por lo que junto con R_{IC} y R_{SS} determinan el tiempo de

carga de C_{HOLD} hasta que alcanza la magnitud del voltaje presente en la fuente de señal analógica.

Con la finalidad que el resultado de la conversión presente un nivel aceptable de precisión, es necesario que en el tiempo de adquisición, que no debe ser mayor de unos 12.5μS, el capacitor se cargue al valor presente en la entrada, por lo que se recomienda que la impedancia de salida de la fuente de la señal analógica no sobrepase los 2.5kΩ.

Finalmente, el concatenamiento de los registros ADRESH:ADRESL sirve para almacenar los diez bits resultantes de la conversión A/D, esto ocurre cuando la conversión se ha completado, produciendo además que el bit GO/$\overline{\text{DONE}}$ del registro ADCON0 se reestablezca, y al mismo tiempo que el bit de la bandera de interrupción, ADIF se active.

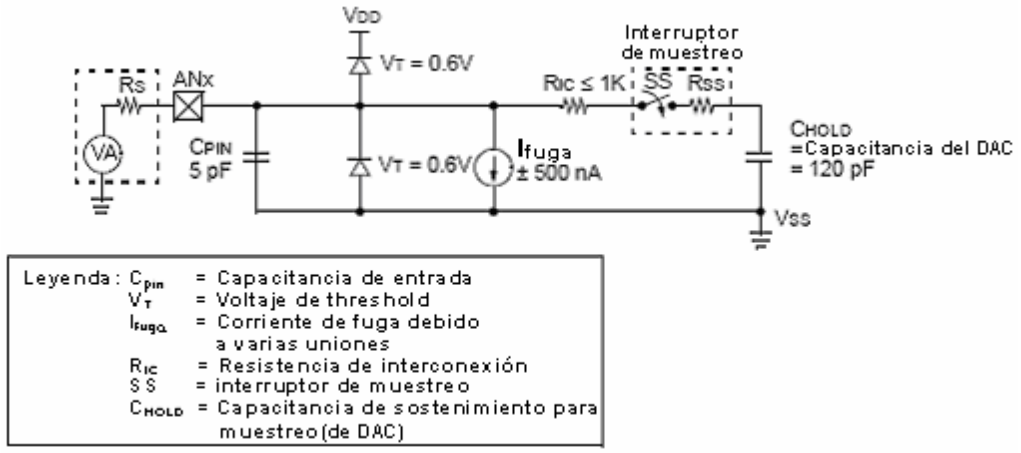


Figura 4.11 Modelo de entrada de los canales de conversión A/D. [29]

4.3.1.5. SUBSITEMA DE COMUNICACIÓN SERIE.

El modulo USART (Universal Synchronous Asynchronous Receiver Transmitter), también llamado SCI (Serial Communications Interface) puede ser

configurado para operar en modo Asíncrono (full-duplex) o Síncrono (half-duplex). Para su implementación se utilizan los pines RC6 (Tx) y RC7 (Rx) para el modo Asíncrono, junto con los registros mostrados en la tabla 4.8.

Tanto los registros TXSTA, RCSTA y SPBRG, se encargan de programar el modo de operación, así como de gestionar e identificar el estado del subsistema SCI. Los registros TXREG y RCREG son los registros buffer de salida y entrada, respectivamente que contienen el byte a ser transferido.

Nombre	Descripción	Dirección
TXSTA	Registro de control y estado de la transmisión.	98h
RCSTA	Registro de control y estado de la recepción.	18h
SPBRG	Registro generador de Baud Rate	99h
TXREG	Registro de transmisión USART	19h
RCREG	Registro de recepción USART	1Ah

Tabla 4.8 Resumen de registros asociados al módulo SCI.

Cuando el subsistema USART se configura en modo asíncrono, es decir limpiando el bit SYNC del registro TXSTA, se utiliza el formato estándar NRZ, con un bit de inicio, ocho o nueve bits de datos y un bit de paro. El bit de paridad no es soportado por el hardware pero si puede ser implementado por software (almacenándose como un noveno bit de datos). La transmisión se inicia con el bit menos significativo.

Un elemento importante del subsistema es el Generador de Baud Rate interno de 8 bits que se utiliza para obtener frecuencias típicas de transmisión, a partir del oscilador principal del microcontrolador. Funcionalmente el transmisor y el receptor son independientes, pero comparten la misma configuración.

4.3.2. DISEÑO DEL CIRCUITO PRINCIPAL.

Tal como se describe al inicio de ésta sección, el bloque de control está centralizado en dos microcontroladores PIC16F877A los cuales realizan funciones independientes pero a su vez ambos comparten datos de información y estados del sistema, auxiliándose cada uno de circuitos suplementarios que complementan el cometido delegado.

Así, un primer microcontrolador (llamado de gestión y control) se encarga de activar/desactivar el sistema de cara al usuario, operar a voluntad del usuario, apoyándose del circuito de mando y señalización; controlar los actuadores utilizando el circuito de potencia, leer y convertir los valores analógicos provenientes del circuito de acondicionamiento. Transferir los datos digitales de cada conversión así como indicarle el estado operativo del sistema al segundo microcontrolador (llamado de procesamiento y comunicación).

Por su parte, las tareas del segundo microcontrolador, incluyen establecer comunicación serie con el teléfono móvil, utilizando una interfaz, para indicarle al microcontrolador de gestión y control que es valido el inicio de una transacción; en el transcurso de la misma, procesa la información digitalizada de cada conversión (datos en términos flujo volumétrico) y organiza tablas con información expresada en términos de volumen. Envía los datos de volúmenes descargados finales al dispositivo móvil, cuando el microcontrolador de gestión y control le indique la finalización de una transacción.

Para realizar las funciones descritas de ambos microcontroladores, es menester realizar ciertas conexiones eléctricas entre ambos microcontroladores y los circuitos auxiliares. La tabla 4.9 presenta un resumen de las conexiones eléctricas de los periféricos de ambos microcontroladores.

MCU Origen	Tareas	Funciones	Recursos	Contraparte
Gestión y Control	Conversiones A/D	ADC0	AN0	Circuito de acondicionamiento.
		ADC1	AN1	
		ADC3	AN3	
Gestión y Control	Lectura de los dispositivos de Mando	Llave	RB4	Circuito auxiliar dispositivos de mando
		Paro	RB3	
		B1(EV1)	RB2	
		B2(EV2)	RB1	
		B3(EV3)	RB0	
Gestión y Control	Señalización	Alerta	RA2	Circuito de potencia
		EV1	RC0	
		EV2	RC1	
		EV3	RC2	
		Marcha	RC3	
		Paro	RC4	
Gestión y Control	Control Actuadores	EV1	RC0	Circuito de potencia
		EV2	RC1	
		EV3	RC2	
Gestión y Control	Transmisión Resultado ADC	PROML	Puerto D (8 bits)	Puerto B del MCU de Procesamiento y Comunicación
		PROMH	RC7(MSB)	RD1 del MCU de Procesamiento y Comunicación
			RC6(LSB)	RD0 del MCU de Procesamiento y Comunicación
Procesamiento y Comunicación	Comunicación Serie	Rx	RC7	Interfaz Serie
		Tx	RC6	

Tabla 4.9 Resumen de conexiones eléctricas y periféricos utilizados.

Las conexiones del circuito principal se pueden observar en la figura 4.12. Con la finalidad de reducir el diagrama se utilizan buses con su respectiva etiqueta para identificar las líneas que transporta. También se utilizan tres conectores para presentar las conexiones con los circuitos auxiliares. De igual forma se omite el circuito de la fuente de alimentación y el oscilador a cristal de 18.432 MHz.

4.3.3. DISPOSITIVOS DE MANDO Y SEÑALIZACIÓN (CIRCUITOS AUXILIARES)

Los dispositivos de mando y señalización son aquellos elementos involucrados en el dialogo hombre-máquina durante la operación de esta última. La tabla 4.10 ilustra las funciones en las que participan dichos elementos, así como sus características.

Cabe mencionar que estos dispositivos están íntimamente relacionados, a través de una circuitería sencilla, con uno de los microcontrolador del sistema, el encargado de la adquisición de la señal analógica y del dialogo hombre-máquina.

La figura 4.13 y 4.14 presentan los diagramas de los circuitos auxiliares al microcontrolador, correspondientes a la etapa de mando y señalización respectivamente. Así mismo, en este último se presenta la etapa de salida de potencia para la operación de los dispositivos actuadores.

El circuito de mando básicamente es un diseño sencillo, ya que la eliminación de rebotes se realiza por software en el microcontrolador. En la figura 4.11 se detalla hacia que puerto, pin y microcontrolador envía la señal eléctrica correspondiente al evento generado por el usuario.

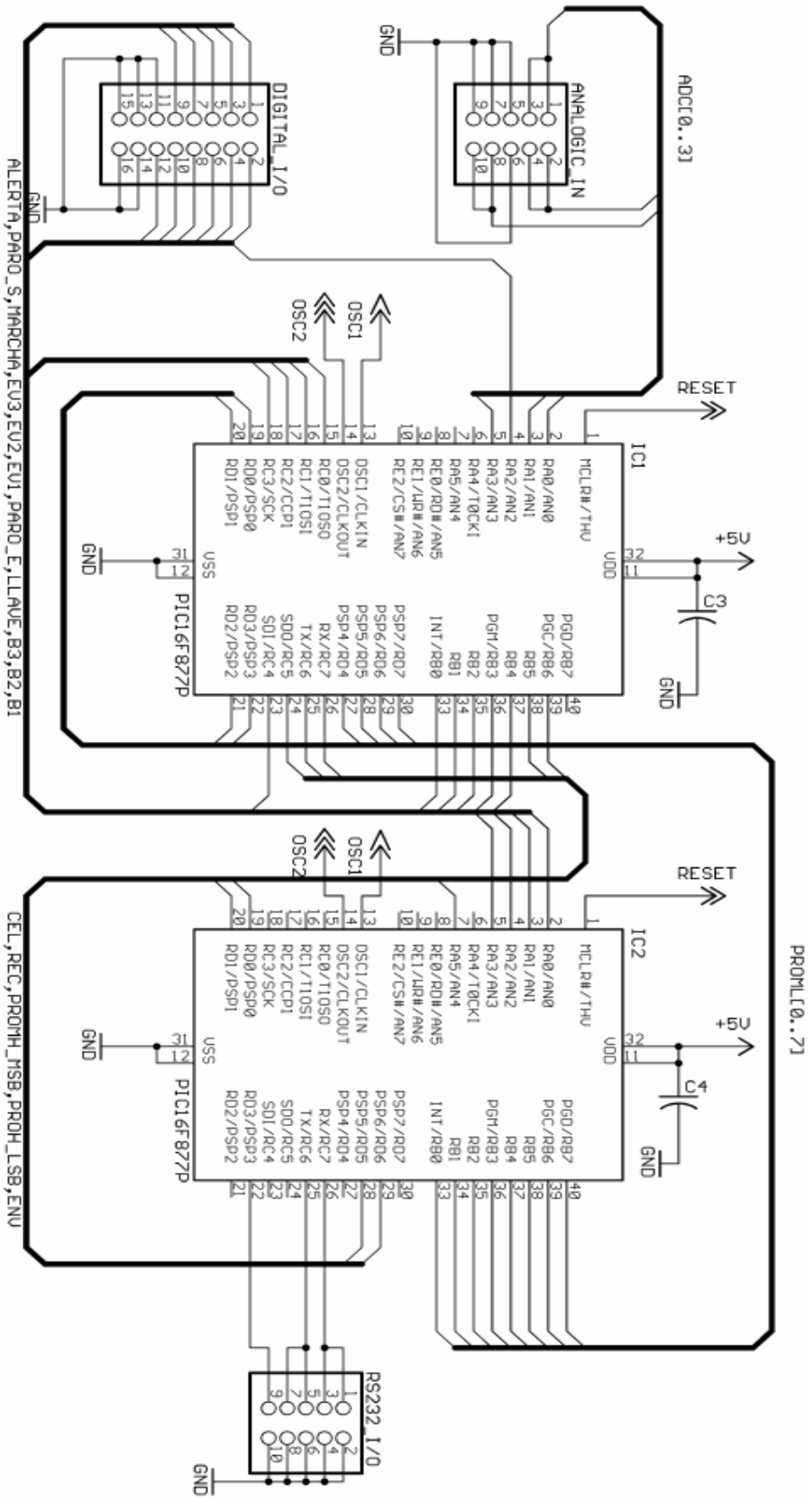


Figura 4.12 Diagrama eléctrico del circuito principal (Microcontroladores)

Función	Descripción	Dispositivo prototipo
Marcha	Interruptor de encendido/apagado del sistema de adquisición.	Conmutador SPST N.A. tipo llave
Paro	Pulsador para detener la descarga de un tipo de combustible en particular.	Pulsador N.A.
EV1	Pulsador para iniciar el servicio de combustible tipo A.	Pulsador N.A
EV2	Pulsador para iniciar el servicio de combustible tipo B.	Pulsador N.A.
EV3	Pulsador para iniciar el servicio de combustible tipo C.	Pulsador N.A
Marcha	Indicador luminoso del estado operativo del sistema.	Led verde
Paro	Indicador luminoso del estado de paro del sistema.	Led verde
Alerta	Indicador luminoso del estado de alerta del sistema.	Led verde
EV1	Indicador luminoso del estado de la electroválvula de salida para el combustible tipo A.	Led Azul
EV2	Indicador luminoso del estado de la electroválvula de salida para el combustible tipo B.	Led Naranja
EV3	Indicador luminoso del estado de la electroválvula de salida para el combustible tipo C.	Led Rojo

Tabla 4.10 Funciones de los dispositivos de mando y señalización.

Por su parte, el circuito auxiliar de señalización aprovecha las salidas de la etapa de potencia para desempeñar su función, esta etapa de potencia toma las señales discretas de los pines RA2 y de RC0 hasta RC4 para amplificarla utilizando el circuito integrado ULN2803A, que es un arreglo de ocho transistores Darlington. Las líneas de control que manejan a los actuadores están protegidas por opto acopladores.

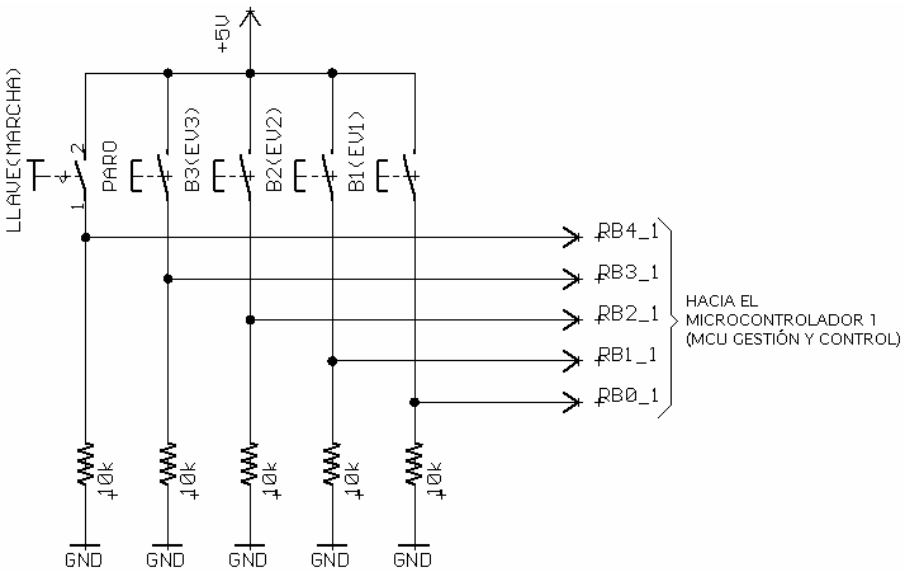


Figura 4.13 Diagrama eléctrico para los dispositivos de mando.

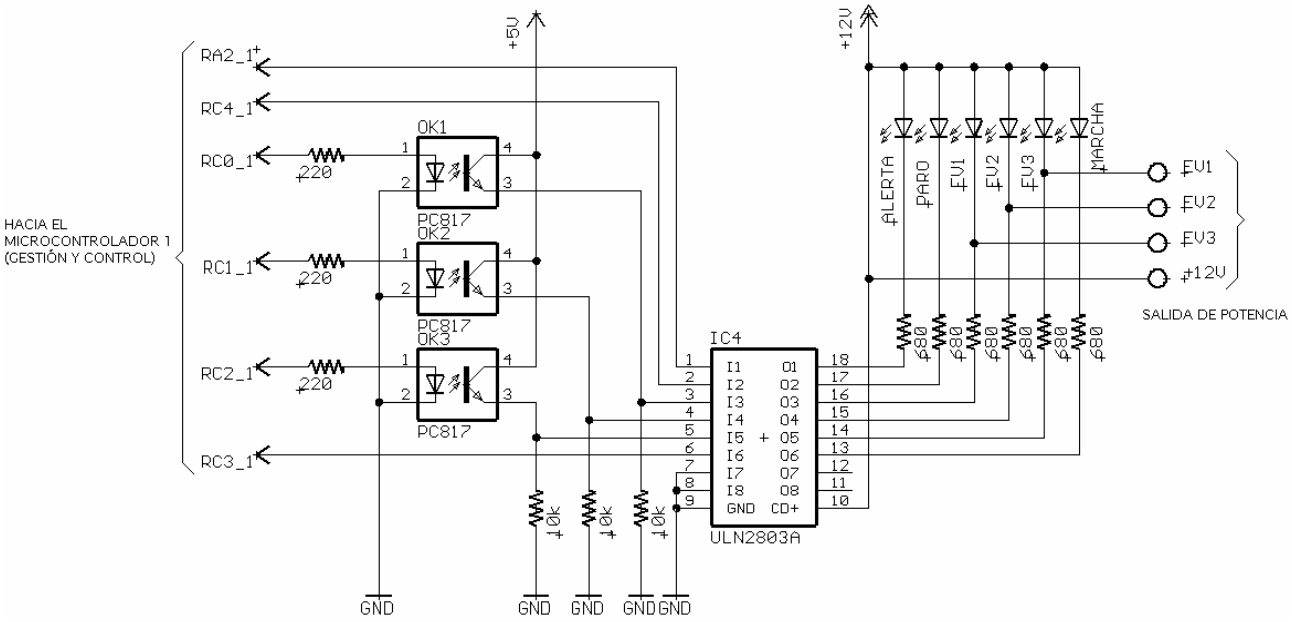


Figura 4.14 Diagrama del circuito auxiliar de potencia y señalización.

4.3.4. SELECCIÓN DE ACTUADORES.

Los actuadores involucrados son las electroválvulas que liberan u obstruyen el paso del líquido a petición del circuito de control, cuando el usuario inicia o finaliza el proceso de descarga del combustible.

Para el diseño del prototipo se utiliza una electroválvula con un solenoide de 12V (de vehículo) que maneje pequeños caudales, siendo ésta excitada desde un circuito de potencia basado en un arreglo octal de transistores Darlington, empaquetados en el ULN2803, con capacidad de entregar hasta 500mA por canal.

Por el tipo de fluido a manipular y sin alterar el circuito de potencia, los aspectos que deberían cumplir las electroválvulas para operar en un sistema real se enlistan en la tabla siguiente.

Factor a considerar	Atributos
Solenoide	12V @ 5.5W (Máx.) Con aislamiento térmico y eléctrico.
Diámetro de acople	5”
Cuerpo de la E.V.	Acero Inoxidable o Aluminio
Cámara y Vástago de la E.V.	Resistencia química al flujo de hidrocarburos.

Tabla 4.11 Características de las electroválvulas para un sistema industrial.

4.4. BLOQUE DE TELECOMUNICACIÓN: TELÉFONO GSM MÓVIL.

Las características de este dispositivo son determinantes no solo en lo que respecta a hardware, si no que también para el desarrollo del MIDlet. Éste bloque se encarga de informar al sistema de instrumentación y control el inicio de una nueva transacción, así como de recibir al final de la misma los datos de volúmenes totales descargados.

Por último encapsula dicha información junto con otra necesaria para la base de datos dentro de un mensaje de texto, y lo envía a través de la red celular GSM del operador seleccionado, con el propósito de reportarle los detalles de la transacción a la estación central.

4.4.1. RECURSOS NECESARIOS.

Los recursos necesarios tanto en hardware como en software que se estiman convenientes y en algunos casos determinados de forma experimental, se presentan en la tabla 4.12.

Requerimiento/Especificación Técnica	**Justificación**
Puerto de conectividad manipulable desde J2ME.	Este puerto se requiere para establecer comunicación con el sistema electrónico de instrumentación y control.
Puerto de carga de batería independiente del puerto de datos.	Se requiere de este modo, para permitirle libertad de carga cuando sea necesario, sin necesidad de retirar el cable de datos.
Teclado sencillo.	Para facilitar al usuario la operación de la aplicación Java.
Pantalla de 128x128 píxeles monocromática.	Para una presentación elegante de la aplicación.
Conectividad local (USB/IrDA/Bluetooth)	Con el objetivo de transferir desde una PC la aplicación al teléfono móvil.
Herramientas de desarrollo para PC.	Necesario en la simulación del software durante la etapa de desarrollo.
Sistema Operativo amigable	Para facilitar al usuario la operación de la aplicación Java.
CLDC 1.1	Necesario para manipular puertos COMM
MIDP 2.0	Necesario para manipular puertos COMM
API para el manejo de SMS.	Requerido para manipular el envío y recepción de SMS.
API para interfases gráficas de usuario.	Para una presentación elegante de la aplicación.
50 kB para almacenar el archivo JAR.	Espacio mínimo para almacenar el MIDlet.

Tabla 4.12 Requerimientos necesarios del teléfono GSM móvil.

4.4.2. EL TELÉFONO MÓVIL NOKIA 5140.

El teléfono Nokia 5140 combina características avanzadas multimedia junto con capacidades de transferencia de datos de alta velocidad utilizando tecnología EDGE. Sin duda estas características generales lo hacen un teléfono sobredimensionado para efectuar las tareas encomendadas dentro del proyecto.

Lastimosamente, dos características importantes para el desarrollo del proyecto, como lo son la versión del CLDC/MIDP y la API para SMS, están disponibles únicamente en dispositivos con características como las que dispone el teléfono seleccionado. La tabla siguiente, presenta los aspectos técnicos de interés, relativos al Nokia 5140.

Requerimiento/Especificación Técnica	**Atributo**
Puerto de conectividad manipulable desde J2ME:	Puerto Infrarrojo
Puerto para carga de batería independiente del puerto de datos:	Si
Teclado:	21 teclas de fácil uso.
Pantalla:	128x128 píxeles, color de 12 bits
Conectividad local:	USB / IrDA
Herramientas de desarrollo para PC:	Si, disponibles en: www.forum.nokia.com
Sistema Operativo:	Nokia OS
Versión de la configuración:	CLDC 1.1
Versión del Perfil:	MIDP 2.0
API para el manejo de SMS:	Wireless Messanging API (JSR-120)
API para interfases gráficas de usuario:	Nokia UI API
Tamaño máximo para archivos JAR:	125kB

Tabla 4.13 Especificaciones técnicas del Nokia 5140. [31]

4.5. INTERFAZ DE COMUNICACIÓN: SISTEMA ELECTRÓNICO Y TELÉFONO GSM MÓVIL.

La interfaz de comunicación es el circuito electrónico que se utiliza para hacer un acople de las características eléctricas entre las señales de comunicación del sistema electrónico de instrumentación y control, y el teléfono GSM del equipo móvil logrando con esto, que ambos bloques hablen el mismo lenguaje.

4.1.1. DISEÑO DE LA INTERFAZ DE COMUNICACIÓN.

Tal como se ha descrito anteriormente, una comunicación de tipo infrarroja en modalidad SIR, requiere de tres componentes principales: una UART compatible, un codificador y un dispositivo transeceptor. La figura 4.16 muestra el esquema general de la interfaz utilizada en el proyecto.

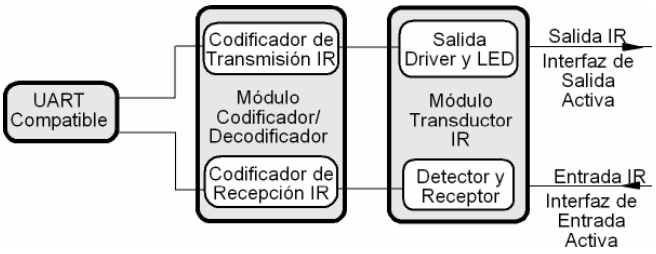


Figura 4.16 Diagrama de bloques de la interfaz. [16]

La UART compatible la ofrece el módulo de comunicación del PIC16F877 utilizado para la comunicación del sistema. Éste subsistema proporciona tres líneas (Tx, Rx y GND) para establecer la comunicación, así como una trama típica con un bit de inicio, ocho bits de datos y bit de paro; con un esquema de codificación NRZ.

En cuanto al módulo de codificación/decodificación infrarroja, la empresa Microchip ofrece el dispositivo MCP2120, que entre otras características brinda compatibilidad con los microcontroladores PIC de gama media, compatibilidad con los tranceivers de la serie TFDS 4xxx / 6xxx de la Vishay Telefunken, soporte de la versión 1.3 de las especificaciones de la capa física de IrDA y selección de la

velocidad de transmisión a través de hardware o software hasta una velocidad máxima de 115.2 kbps.

Finalmente, el dispositivo transceptor elegido es el TFDS 6000 que ofrece soporte tanto para velocidades SIR y FIR, así como el cumplimiento de la versión 1.3 del estándar de IrDA.

La figura siguiente presenta el diagrama eléctrico de la interfaz desarrollada, la cual opera a una velocidad de 57.6 kbps.

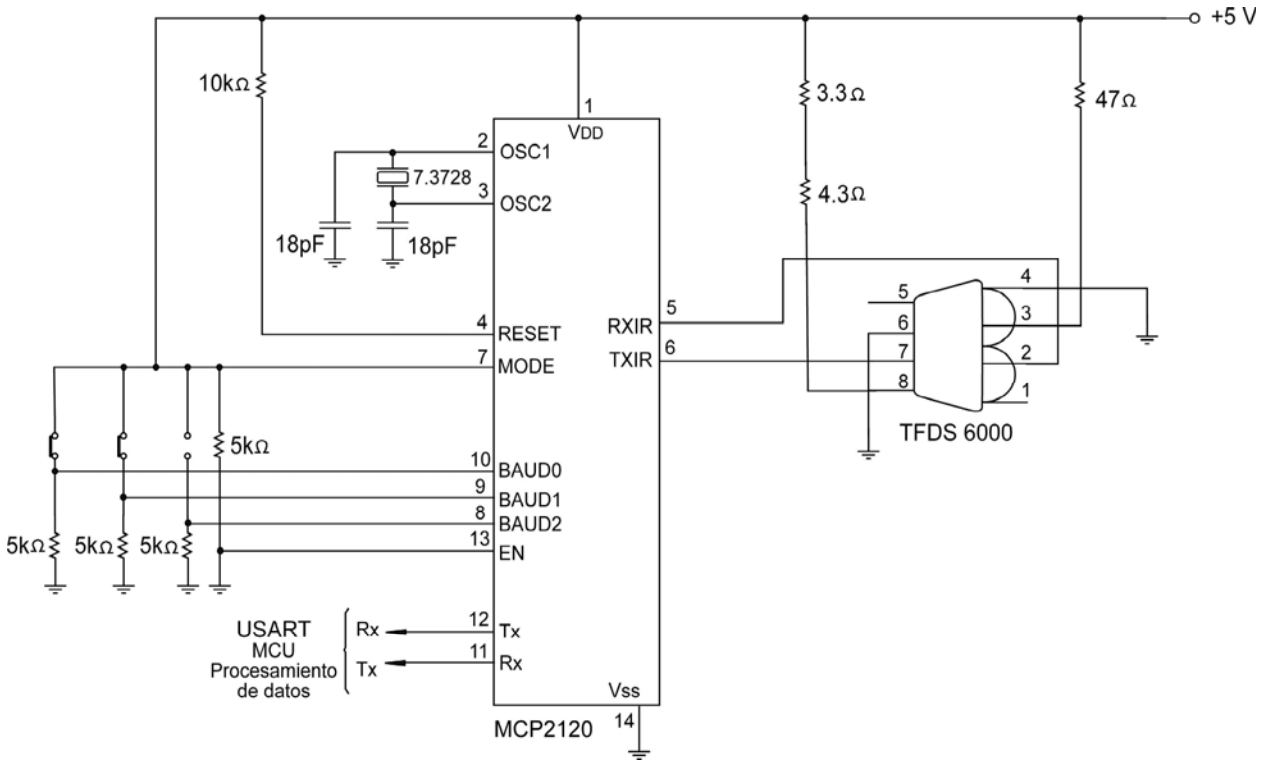


Figura 4.17. Diagrama eléctrico de la interfaz IrDA construida.

4.6. BLOQUE DE LA ESTACIÓN CENTRAL.

La estación central es la encargada de recibir la información de cada transacción realizada para su posterior organización en tablas dentro de una base de datos.

Para ellos se consideran dos elementos importantes, el teléfono dedicado, que se encarga de la recepción de los SMS enviados por los equipos móviles, así como de transmitir un acuse de recibido al equipo móvil que origino la comunicación.

El segundo elemento, es la computadora central que por medio de software gestiona el desempaquetado de la información del SMS entrante, así como su manipulación para llenar las tablas de la base de datos y finalmente se encarga de formar el mensaje Ack saliente.

Para lograr comunicación entre estos dos elementos, es necesario contar con el cable de datos del teléfono seleccionado, junto con los drivers y el software de comunicación proporcionado por el fabricante.

4.6.1. COMPUTADORA CENTRAL.

El la computadora central residirá el software administrativo desarrollado bajo Microsoft Visual Basic 6.0. Por lo que, para efectuar las tareas encomendadas es necesario que la computadora cuente con los recursos que se enlistan en la tabla 4.14.

El control Active Xperts SMS and Pager Toolkit 4.1, es un control descargable de internet de forma gratuita y faculta a que el software desarrollado con Visual Basic, manipule a traves del puerto USB o COMM (dependerá del teléfono seleccionado) SMS entrantes o salientes.

Recurso	Atributo
Microprocesador	Pentium III
RAM	256 Mb
Puerto de comunicación	USB/Serie
Sistema Operativo	Microsoft Windows Me
Software Gestor de Hojas de Calculo	Microsoft Excel XP
Navegador Web	Internet Explorer 4.x o superior
Controles	Active Xperts SMS and Pager Toolkit 4.1
	Microsoft Excel 10.0
	Microsoft ActiveX data 2.8

Tabla 4.14 Recursos necesarios en la computadora de la estación central.

4.6.2. TELÉFONO GSM DE LA ESTACIÓN CENTRAL.

Ya que el control Active Xperts SMS and Pager Toolkit 4.1, tiene la capacidad de manipula los mensajes de texto entrantes y salientes, es necesario recurrir a la lista de teléfonos sobre los cuales el control tiene efecto, para su elección.

Entre el listado de teléfonos, se elige un dispositivo GSM Siemens S65, ya que son de gran popularidad en el mercado, al igual que sus herramientas de comunicación para PC.

Éste teléfono es “dedicado” únicamente para recibir los mensajes de texto provenientes de equipos móviles que se encuentren registrados, por lo que, el software administrativo utiliza el numero telefónico asignado y el IMEI con el objetivo de comparar si el SMS es de una equipo valido o no.

CAPÍTULO V: *DISEÑO DEL SOFTWARE*

5.1. PANORAMA GENERAL DEL SOFTWARE DE LA APLICACIÓN.

El software involucrado en el funcionamiento del proyecto, se puede dividir en tres secciones diferentes, ya que se dispone de tres dispositivos programables dentro de la arquitectura del mismo.

Es así que se tiene un software de bajo nivel ejecutándose en los microcontroladores PIC del sistema electrónico de instrumentación y control. Por otra parte, el teléfono GSM del equipo móvil, ejecuta una aplicación orientada a objetos bajo la plataforma J2ME. Y finalmente en la estación central, el software administrativo orientado a eventos desarrollado bajo Microsoft Visual Basic.

En el transcurso del capítulo, se describen los aspectos considerados, junto con el flujograma que presenta la solución implementada a las tareas encomendadas a cada dispositivo. Se presenta el código de cada programa en la sección de apéndices.

5.2. DISEÑO DEL SOFTWARE DE BAJO NIVEL.

El software de bajo nivel reside en los dos microcontroladores que gobiernan el sistema electrónico del proyecto. Los microcontroladores PIC poseen un núcleo con arquitectura Harvard basado en un procesador RISC. Bajo éstas características ofrecen un set reducido de 35 instrucciones de 14 bits, logrando con ello que cada instrucción consuma únicamente un ciclo de instrucción y dos ciclos para las instrucciones de salto.

A continuación se presentan los aspectos considerados que condicionan el modo de funcionamiento del software y el flujograma que da solución a las tareas encomendadas y que fundamenta la secuencia de instrucciones codificadas en lenguaje PIC.

5.2.1. SOFTWARE DEL MCU DE GESTIÓN Y CONTROL.

5.2.1.1. CONSIDERACIONES.

El software residente en el microcontrolador encargado de la gestión y control, debe realizar las siguientes tareas:

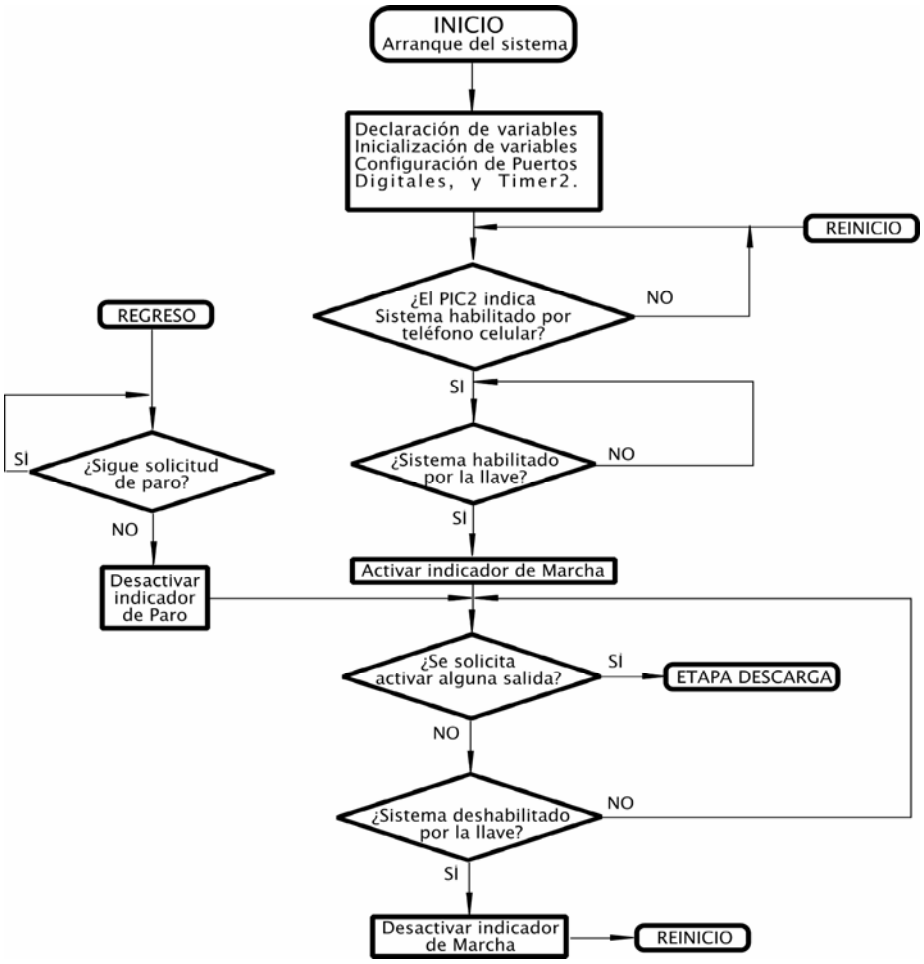
- Habilitar el circuito de mando y señalización para iniciar una transacción.
- Habilitar la electroválvula de descarga para el combustible seleccionado por el usuario.
- Leer el canal de conversión A/D relacionados a la electroválvula que está en funcionamiento.
- Transferir, mientras dure el proceso de descarga, los resultados de las conversiones al microcontrolador de procesamiento y comunicaciones.
- Detectar el fin del proceso de descarga y esperar si se habilita otra electroválvula o bien finalizar la transacción.

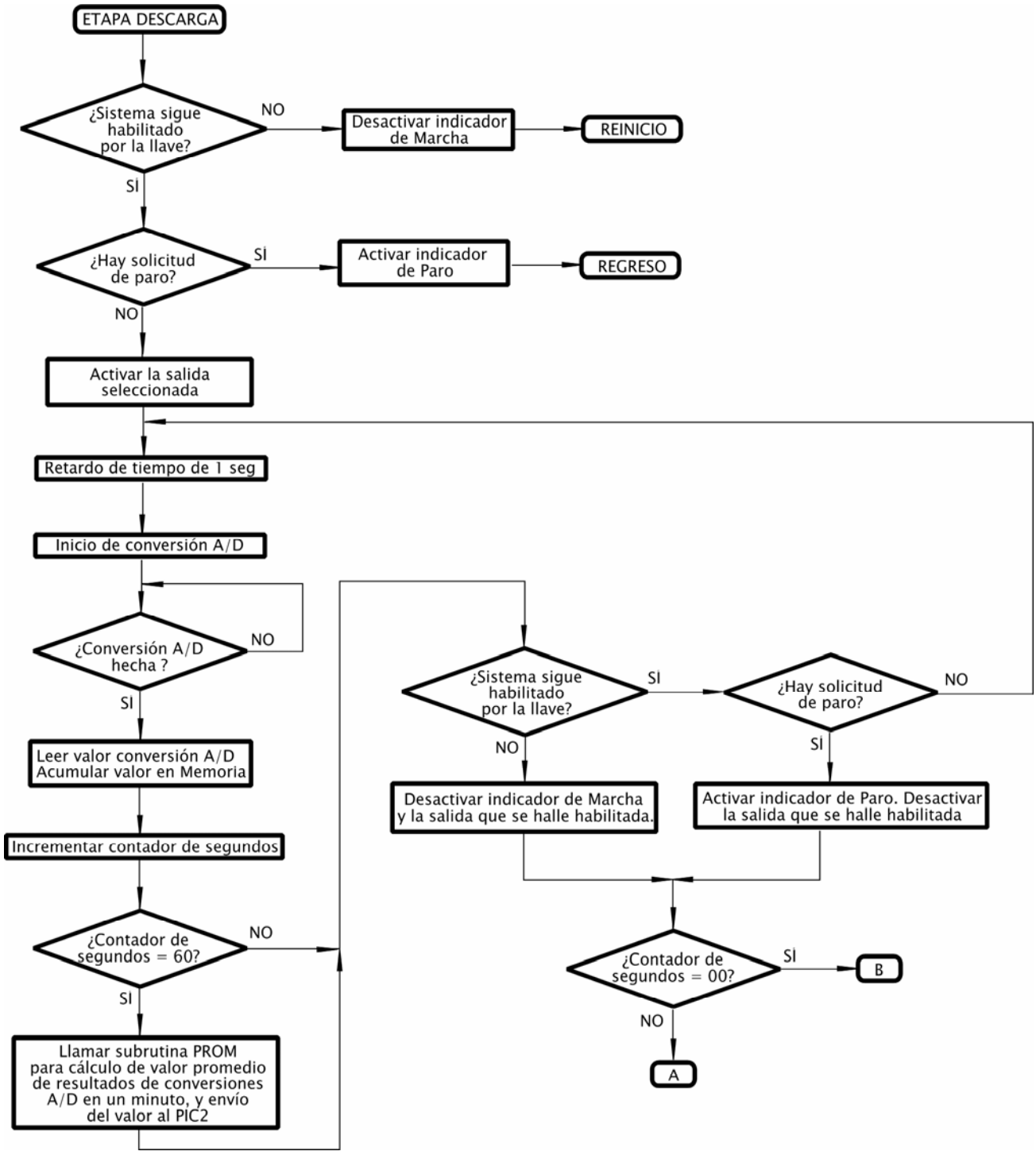
Para realizar las tareas destinadas al microcontrolador, el algoritmo presentado como solución ejecuta los siguientes pasos:

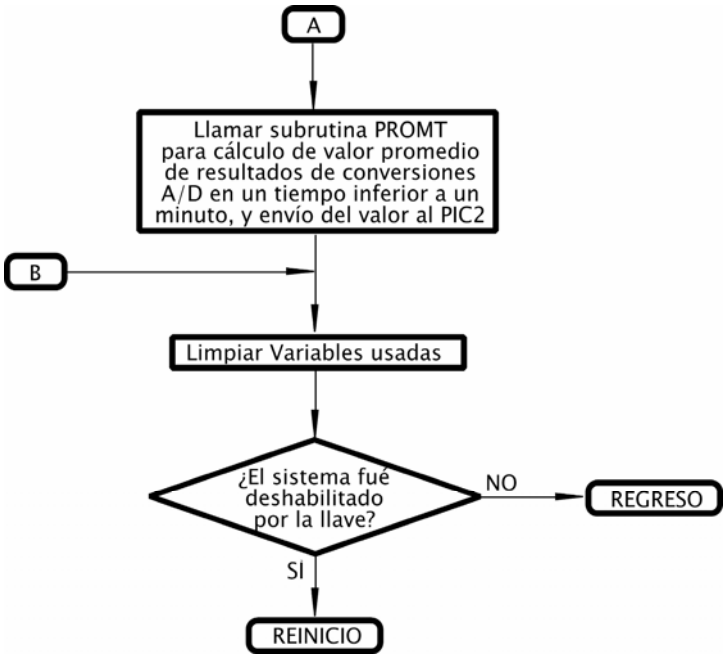
- Esperar que el otro MCU indique cuando un usuario ha sido validado por parte del teléfono móvil, para echar andar el sistema.
- Esperar que el usuario validado accione la llave de arranque del sistema y alguna de las salidas.
- Cuando el usuario presione alguno de los botones para activar alguna de las salidas, el MCU manda activar la salida correspondiente.
- Posteriormente a la activación de la salida, el MCU estará muestreando cada segundo un valor de voltaje dado por el sensor de flujo en dicha salida, mediante su módulo A/D. Al completar cada minuto, el MCU calcula el promedio de los valores muestreados, y envía dicho valor junto con un dato que representa la salida activada al MCU de Procesamiento.

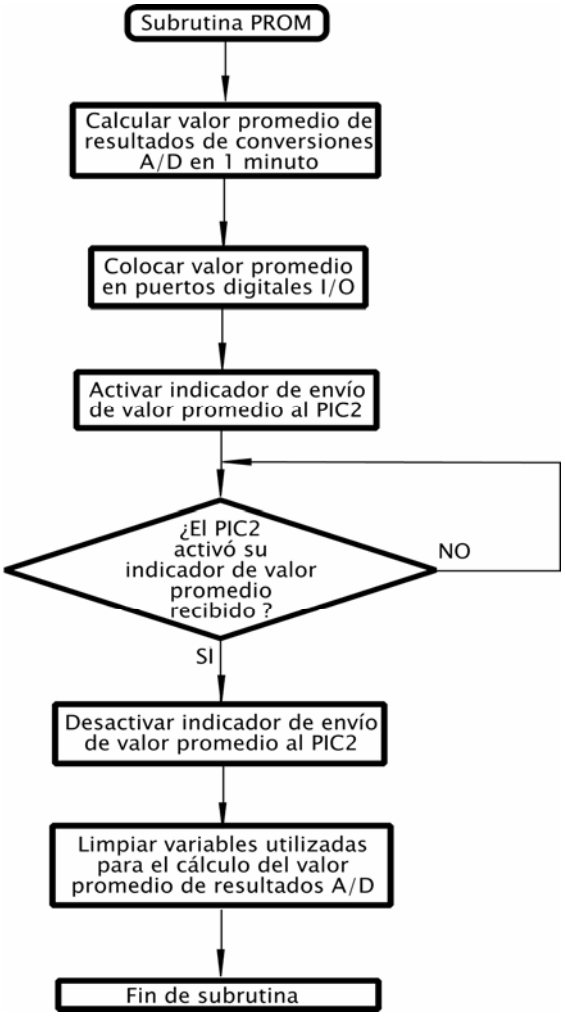
- Mientras la descarga del líquido continúa y efectúa regularmente el proceso anterior, verifica si se ha presionado ya sea el botón de Paro o se ha liberado la llave del sistema.
- Para el caso en que se ha presionado Paro, el MCU manda desactivar la salida abierta, calcula el promedio de la cuenta que se llevaba junto con el dato de la salida y lo envía al otro MCU para su almacenamiento. Después regresa a esperar por la activación de cualquier otro botón.
- Si se libera la llave durante una descarga se calcula el promedio y envía los datos al otro MCU.
- Luego del paso anterior o para el caso en el que la llave se libera mientras no hay ninguna descarga, este MCU limpia sus variables y se reinicia, esperando nuevamente que el MCU de Procesamiento le notifique un usuario válido.

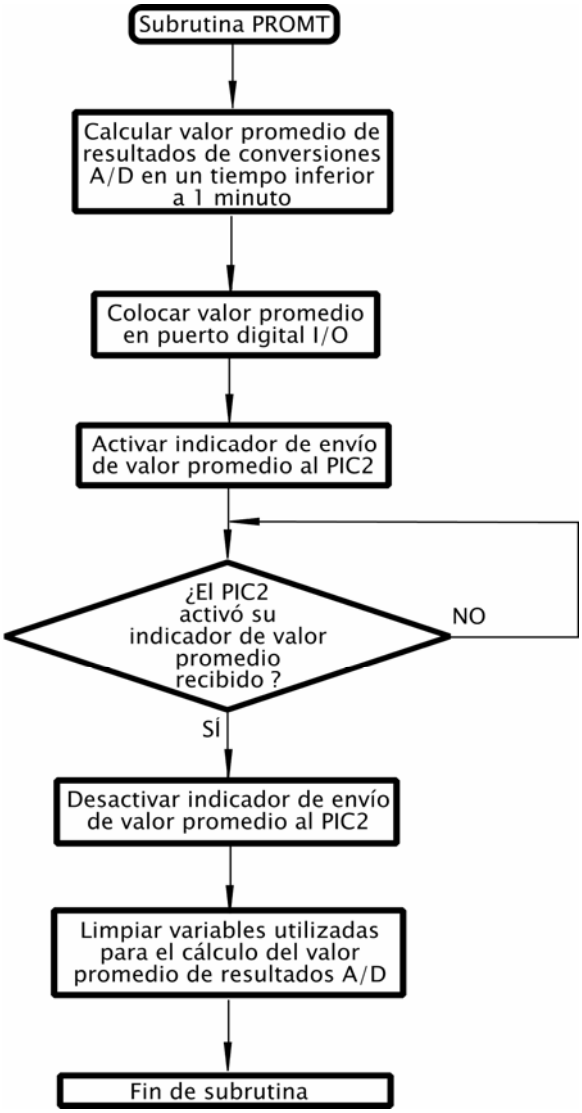
5.2.1.2. FLUJOGRAMA.




ETAPA DESCARGA
¿Sistema sigue habilitado por la llave?
NO
Desactivar indicador de Marcha
REINICIO
SI
¿Hay solicitud de paro?
SI
Activar indicador de Paro
REGRESO
NO
Activar la salida seleccionada
Retardo de tiempo de 1 seg
Inicio de conversión A/D
¿Conversión A/D hecha ?
NO
SI
Leer valor conversión A/D Acumular valor en Memoria
Incrementar contador de segundos
¿Contador de segundos = 60?
NO
SI
Llamar subrutina PROM para cálculo de valor promedio de resultados de conversiones A/D en un minuto, y envío del valor al PIC2
¿Sistema sigue habilitado por la llave?
SI
¿Hay solicitud de paro?
NO
SI
NO
Desactivar indicador de Marcha y la salida que se halle habilitada.
Activar indicador de Paro. Desactivar la salida que se halle habilitada
¿Contador de segundos = 00?
SI
B
NO
A


A
Llamar subrutina PROMT para cálculo de valor promedio de resultados de conversiones A/D en un tiempo inferior a un minuto, y envío del valor al PIC2
B
Limpiar Variables usadas
¿El sistema fué deshabilitado por la llave?
NO
REGRESO
SI
REINICIO


Subrutina PROM
Calcular valor promedio de resultados de conversiones A/D en 1 minuto
Colocar valor promedio en puertos digitales I/O
Activar indicador de envío de valor promedio al PIC2
¿El PIC2 activó su indicador de valor promedio recibido ?
NO
SI
Desactivar indicador de envío de valor promedio al PIC2
Limpiar variables utilizadas para el cálculo del valor promedio de resultados A/D
Fin de subrutina



5.2.2. SOFTWARE DEL MCU DE PROCESAMIENTO Y COMUNICACIONES.

5.2.2.1. CONSIDERACIONES.

El microcontrolador de Procesamiento y Comunicaciones está destinado a realizar las siguientes actividades:

- Esperar a que el teléfono establezca comunicación para indicarle el inicio de una transacción.
- Comunicarle dicho estado al microcontrolador de gestión y control.

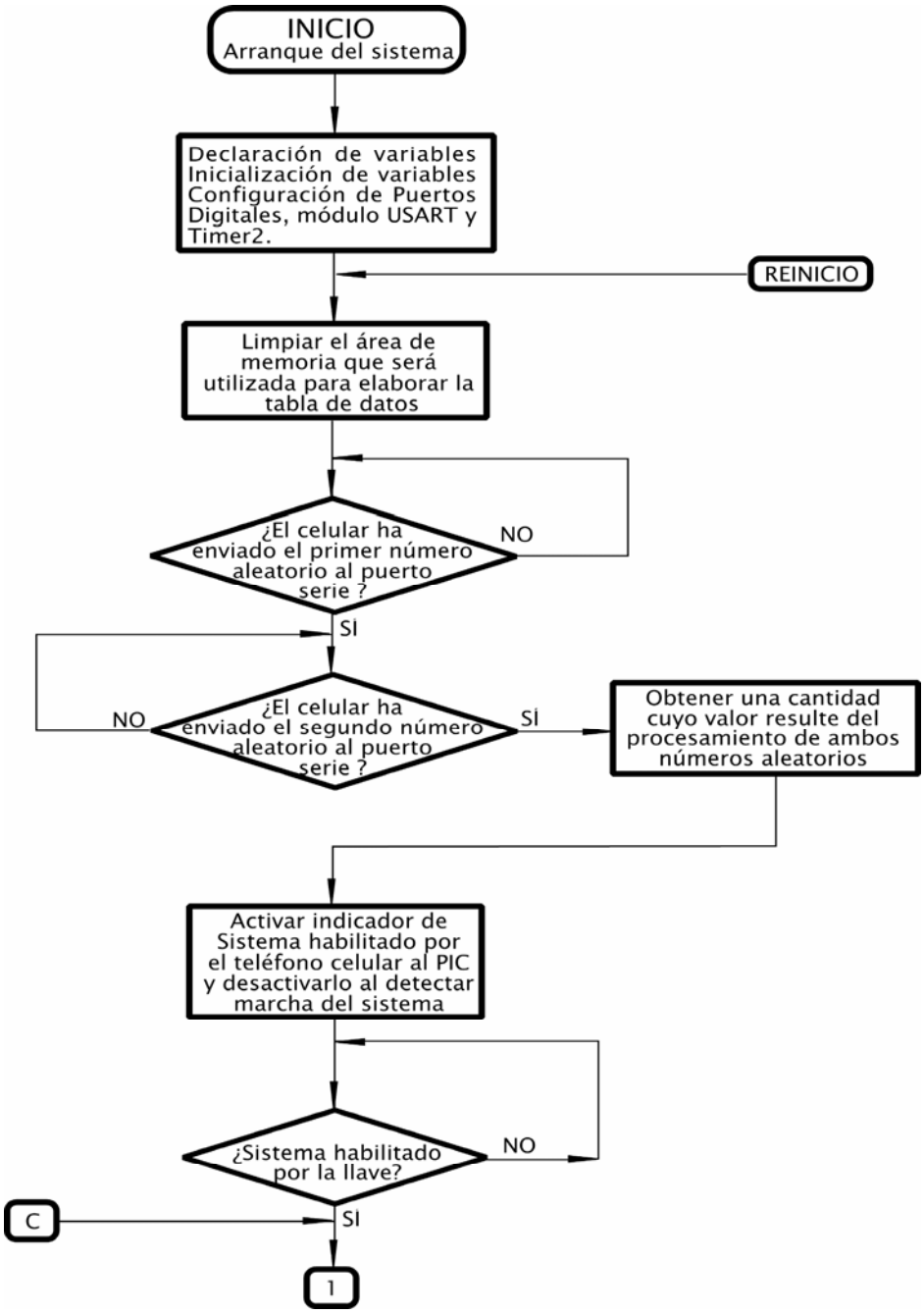
- Esperar los datos de las conversiones A/D procedentes del microcontrolador de gestión y control.
- Procesar dicha información para obtener valores útiles y almacenarlos.
- Cuando el microcontrolador de gestión y control le comunique que la transacción a finalizado, debe gestionar la comunicación para transferir los volúmenes totales al teléfono GSM del equipo móvil.

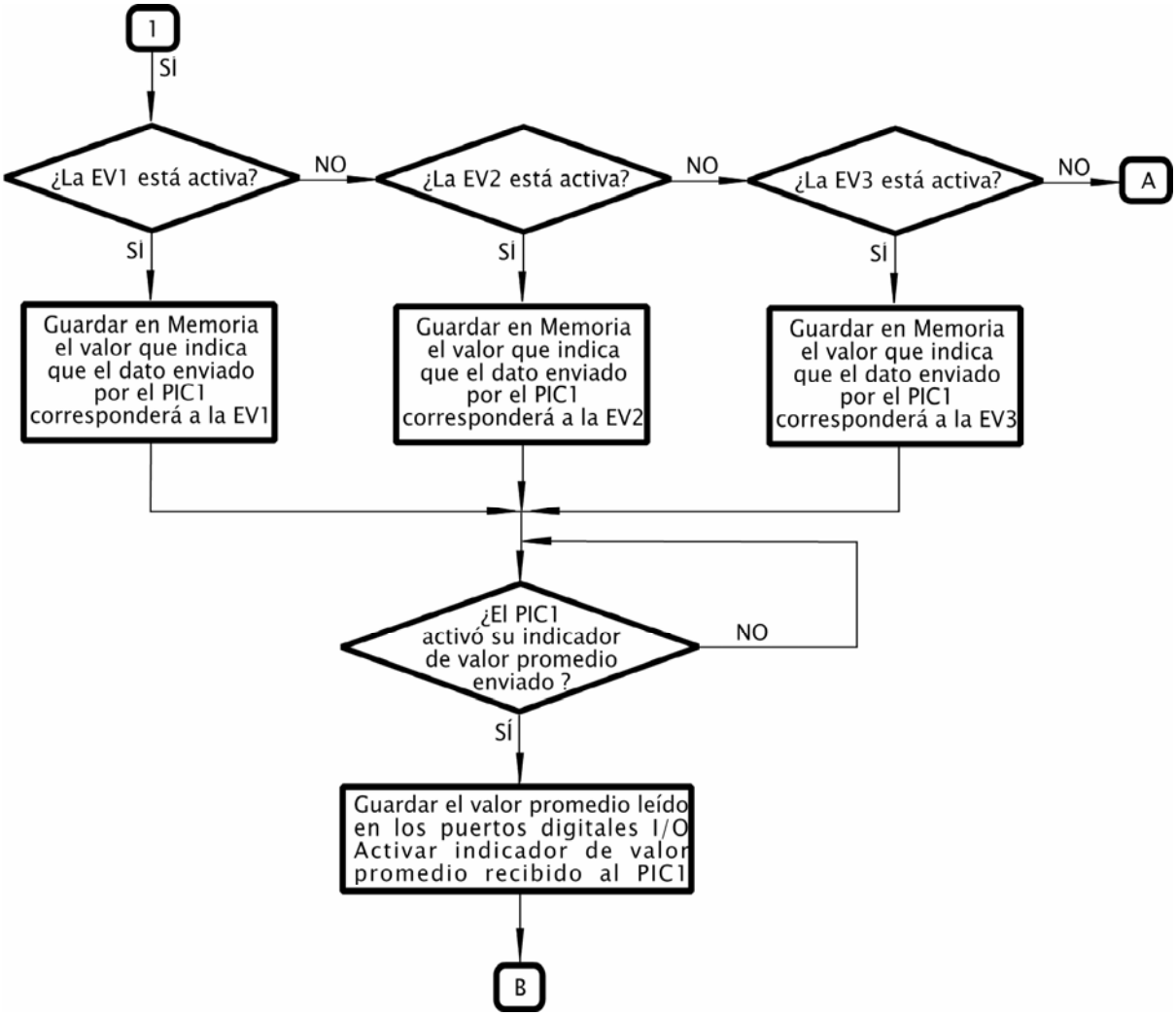
La secuencia de pasos considerados como validos que prestan solución a las actividades por desempeñarse por parte del software de éste microcontrolador, se enlistan a continuación:

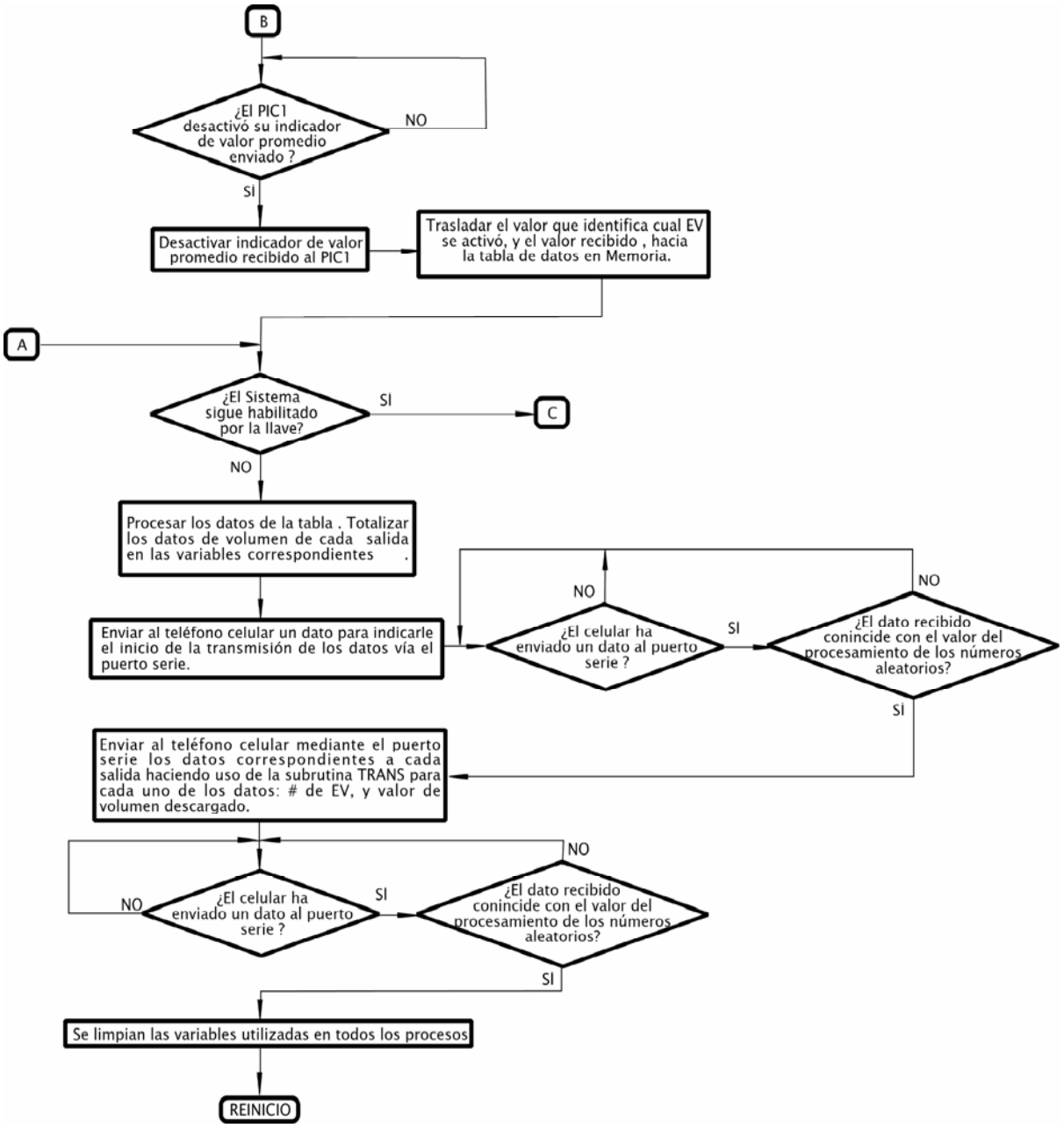
- Esperar que el teléfono móvil le indique cuando un usuario ha sido validado, para echar andar el sistema. Para ello, esperará recibir dos datos aleatorios generados por el mismo móvil.
- Obtener un valor como resultado del procesamiento de ambos números aleatorios.
- Indicar al MCU de Gestión y Control que el teléfono móvil ha permitido el acceso al sistema, para que dicho MCU proceda a iniciar la marcha.
- Esperar a recibir los datos del otro MCU enviados cada minuto en los que detalla el valor promedio de la descarga y la salida a la que corresponde dicho valor. Luego de recibir cada dato, lo almacena en memoria.
- Al detectar que la llave del sistema ha sido liberada, esperar por los últimos datos enviados si los hay.
- Luego de que la llave ha sido removida, procesar los datos almacenados, de tal forma que los valores obtenidos correspondan al equivalente del volumen descargado para cada salida.
- Enviar un dato cualquiera al móvil para indicarle que el MCU está listo para enviarle los datos totales.
- Esperar recibir un dato del teléfono móvil. Este dato deberá coincidir con el resultado del procesamiento de ambos números aleatorios recibidos al inicio.
- Transmitir toda la información al móvil.

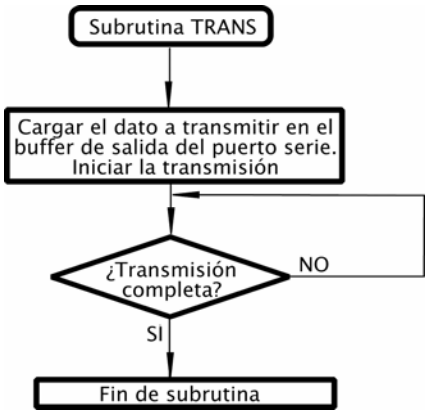
- Luego de finalizar la transmisión de los datos, esperar que el móvil envíe nuevamente el resultado de los números aleatorios, como indicación que este MCU ya puede borrar sus datos y reiniciarse y comenzar un nuevo proceso.

5.2.2.2. FLUJOGRAMA.




1
SI
¿La EV1 está activa?
NO
¿La EV2 está activa?
NO
¿La EV3 está activa?
NO
A
SI
SI
SI
Guardar en Memoria el valor que indica que el dato enviado por el PIC1 corresponderá a la EV1
Guardar en Memoria el valor que indica que el dato enviado por el PIC1 corresponderá a la EV2
Guardar en Memoria el valor que indica que el dato enviado por el PIC1 corresponderá a la EV3
¿El PIC1 activó su indicador de valor promedio enviado ?
NO
SÍ
Guardar el valor promedio leído en los puertos digitales I/O Activar indicador de valor promedio recibido al PIC1
B


B
¿El PIC1 desactivó su indicador de valor promedio enviado ?
NO
SI
Desactivar indicador de valor promedio recibido al PIC1
Trasladar el valor que identifica cual EV se activó, y el valor recibido , hacia la tabla de datos en Memoria.
A
¿El Sistema sigue habilitado por la llave?
SI
C
NO
Procesar los datos de la tabla . Totalizar los datos de volumen de cada salida en las variables correspondientes .
Enviar al teléfono celular un dato para indicarle el inicio de la transmisión de los datos vía el puerto serie.
NO
¿El celular ha enviado un dato al puerto serie ?
SI
NO
¿El dato recibido conincide con el valor del procesamiento de los números aleatorios?
SI
Enviar al teléfono celular mediante el puerto serie los datos correspondientes a cada salida haciendo uso de la subrutina TRANS para cada uno de los datos: # de EV, y valor de volumen descargado.
NO
NO
¿El celular ha enviado un dato al puerto serie ?
SI
¿El dato recibido conincide con el valor del procesamiento de los números aleatorios?
SI
Se limpian las variables utilizadas en todos los procesos
REINICIO


Subrutina TRANS
Cargar el dato a transmitir en el buffer de salida del puerto serie. Iniciar la transmisión
¿Transmisión completa?
NO
SI
Fin de subrutina

5.3. DISEÑO DEL MIDLET HERMES.

El MIDlet esta desarrollado bajo CLDC 1.1 y MIDP 2.0 para ser ejecutado en un teléfono Nokia 5140, que ofrece las características explicadas en el capítulo anterior. Dentro de la arquitectura del proyecto, esta aplicación manipula al teléfono móvil para comunicarse con la estación central, con la finalidad de informarle los detalles de la transacción.

Basado en la función a desempeñar, el nombre del MIDlet se debe al personaje mitológico griego que servía de mensajero de los dioses y del comercio.

5.3.1. CONSIDERACIONES.

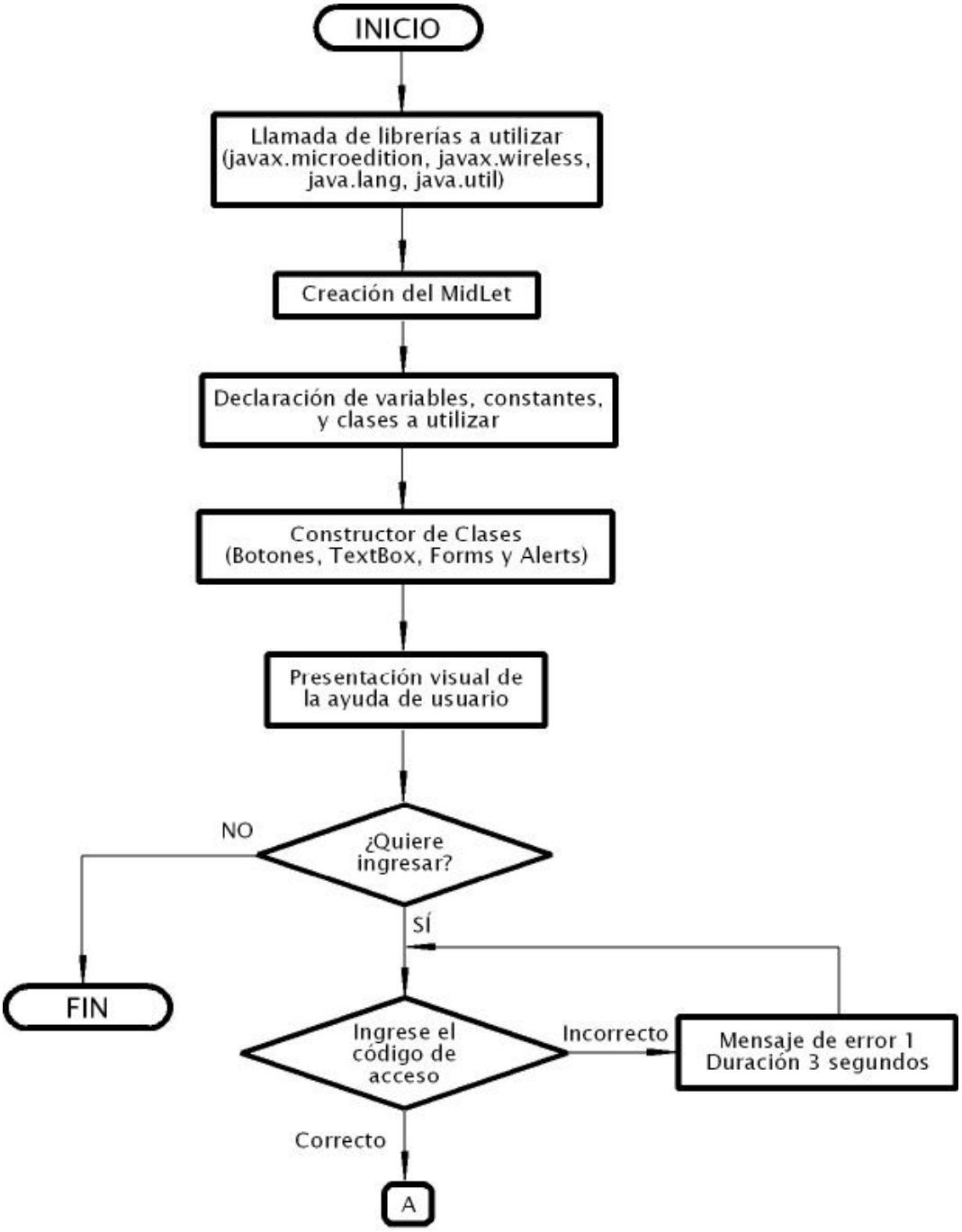
Las siguientes actividades son atribuidas a la aplicación Hermes para su tratamiento:

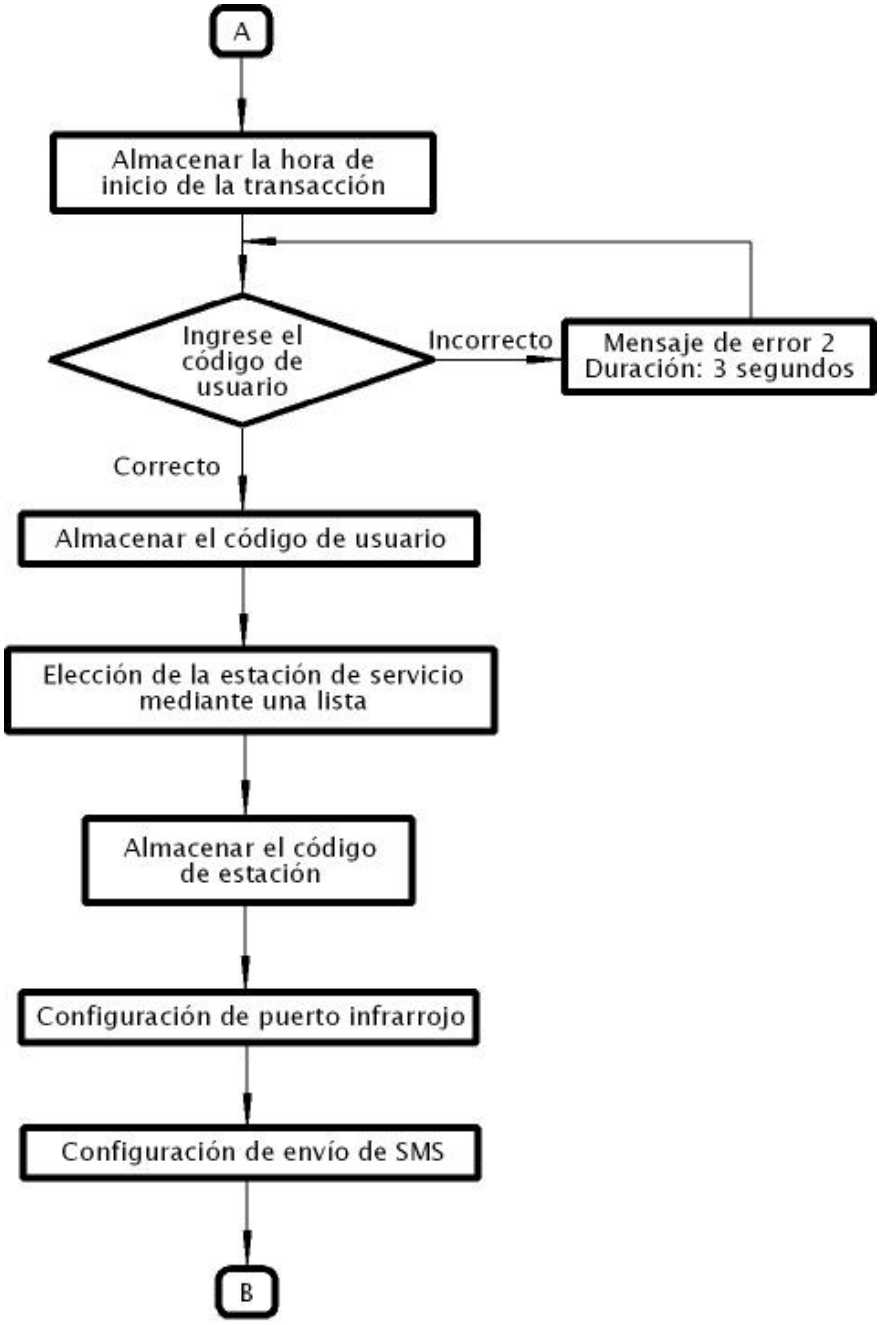
- Comunicarse con el usuario para habilitar el sistema antes de cada transacción.
- Informarle al sistema electrónico de Instrumentación y Control que puede habilitarse para la operación de descarga.
- Esperar la transferencia por parte del sistema electrónico de los volúmenes totales descargados.
- Empaquetar la información junto con otros detalles administrativos en un SMS.
- Enviar el SMS hacia la estación central a través de la red GSM del operador elegido.
- Esperar respuesta por parte de la estación central para dar por finalizada una transacción.

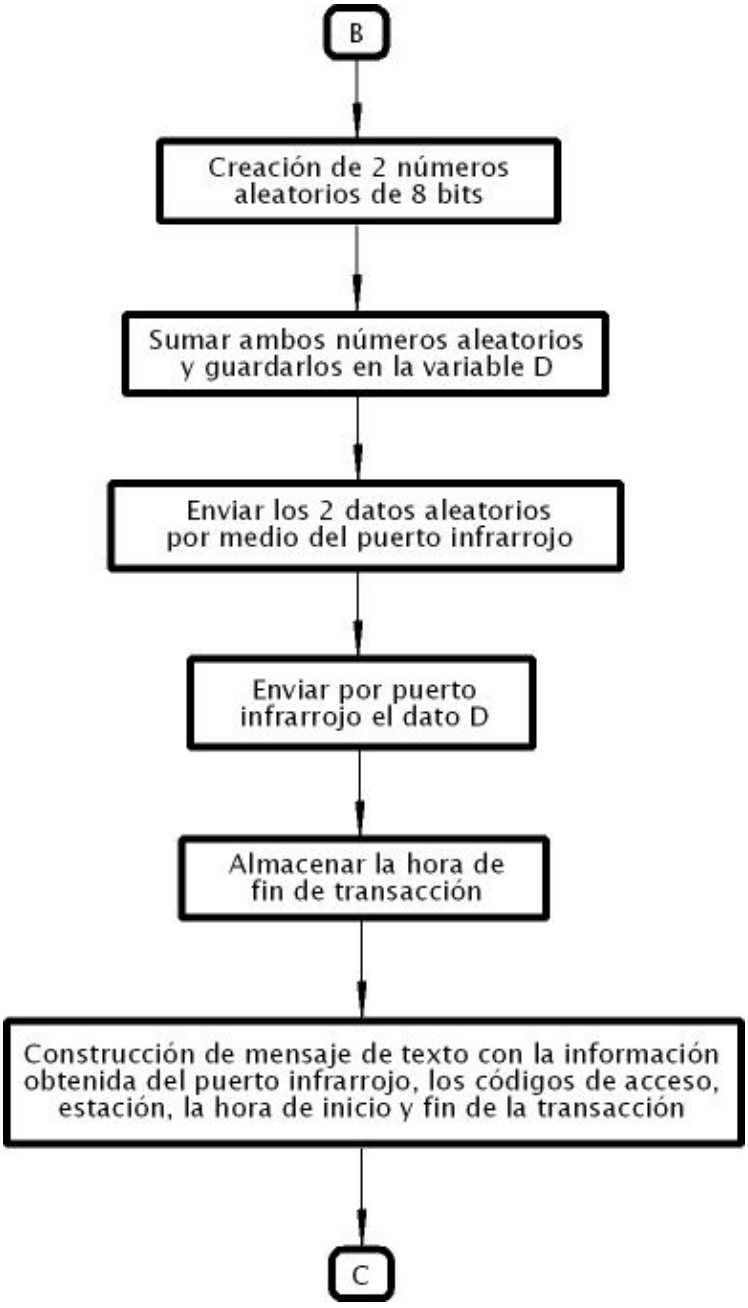
El algoritmo diseñado para satisfacer las tareas antes mencionadas, se detalla a continuación:

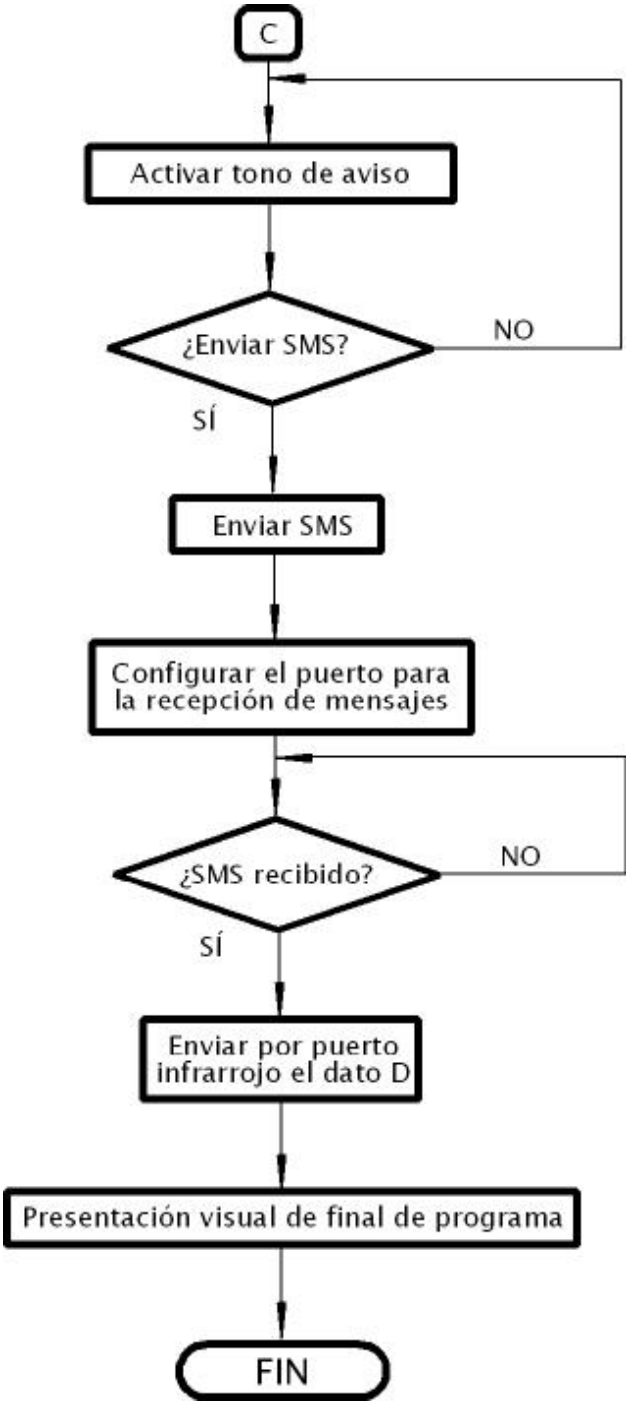
- Presentar las pantallas correspondientes de código de acceso, código de usuario y código de la estación, comparar y guardar los datos obtenidos en variables especificas en la memoria del móvil.
- Obtener la hora del teléfono para guardarla en memoria como hora de inicio de transacción.
- Crear dos datos aleatorios de 8 bits cada uno, los cuales serán enviados al Sistema de Instrumentación y Control, por medio del puerto serie para activar los mismos.
- Operar ambos datos aleatorios y guardar el resultado en una variable especifica en la memoria.
- Esperar que el Sistema Electrónico de Instrumentación y Control envié un dato por medio del puerto infrarrojo para indicar que se iniciará la transmisión de datos de parte del mismo.
- Enviar por el puerto IrDA al Sistema de Instrumentación y Control el resultado de la suma de los dos números aleatorios.
- Esperar los datos de las mediciones realizadas en el Sistema de Instrumentación y Control por el puerto infrarrojo que contiene el tipo de combustible descargado y la descarga realizada; y guardarlas en memoria.
- Obtener la hora del teléfono para guardarla en memoria como hora de finalización de transacción.
- Se pasa a la creación del mensaje de texto o SMS con los datos obtenidos: usuario, estación, tipo de combustible, descarga realizada, hora de inicio de la transacción, finalización de la transición e IMEI.
- Presentar la pantalla para indicar que se va a enviar el SMS.
- Esperar un SMS procedente de la estación base o central indicando que la información ha sido procesada.
- Cuando se recibido el SMS se envía el dato de los dos números aleatorios sumados por medio del puerto serie hacia el Sistema de Instrumentación y Control.
- Presentar la pantalla de finalización de programa.

5.3.2. FLUJOGRAMA.




A
Almacenar la hora de inicio de la transacción
Ingrese el código de usuario
Incorrecto
Mensaje de error 2
Duración: 3 segundos
Correcto
Almacenar el código de usuario
Elección de la estación de servicio mediante una lista
Almacenar el código de estación
Configuración de puerto infrarrojo
Configuración de envío de SMS
B


B
Creación de 2 números aleatorios de 8 bits
Sumar ambos números aleatorios y guardarlos en la variable D
Enviar los 2 datos aleatorios por medio del puerto infrarrojo
Enviar por puerto infrarrojo el dato D
Almacenar la hora de fin de transacción
Construcción de mensaje de texto con la información obtenida del puerto infrarrojo, los códigos de acceso, estación, la hora de inicio y fin de la transacción
C


C
Activar tono de aviso
¿Enviar SMS?
NO
SÍ
Enviar SMS
Configurar el puerto para la recepción de mensajes
¿SMS recibido?
NO
SÍ
Enviar por puerto infrarrojo el dato D
Presentación visual de final de programa
FIN

5.4. SOFTWARE ADMINISTRATIVO.

El software administrativo esta desarrollado en Visual Basic 6.0 junto con los controles Active Xperts SMS and Pager Toolkit 4.1, Microsoft Excel 10.0 y Microsoft ActiveX data 2.8, permitiendo así que la computadora de la estación central, a través del cable de datos del teléfono móvil dedicado, tenga total acceso a las bandejas de entrada y salida de SMS del teléfono y también permitirle el acceso a Microsoft Excel con la finalidad de registrar en una tabla los pormenores de cada transacción.

5.4.1. CONSIDERACIONES.

El software debe realizar las siguientes tareas administrativas:

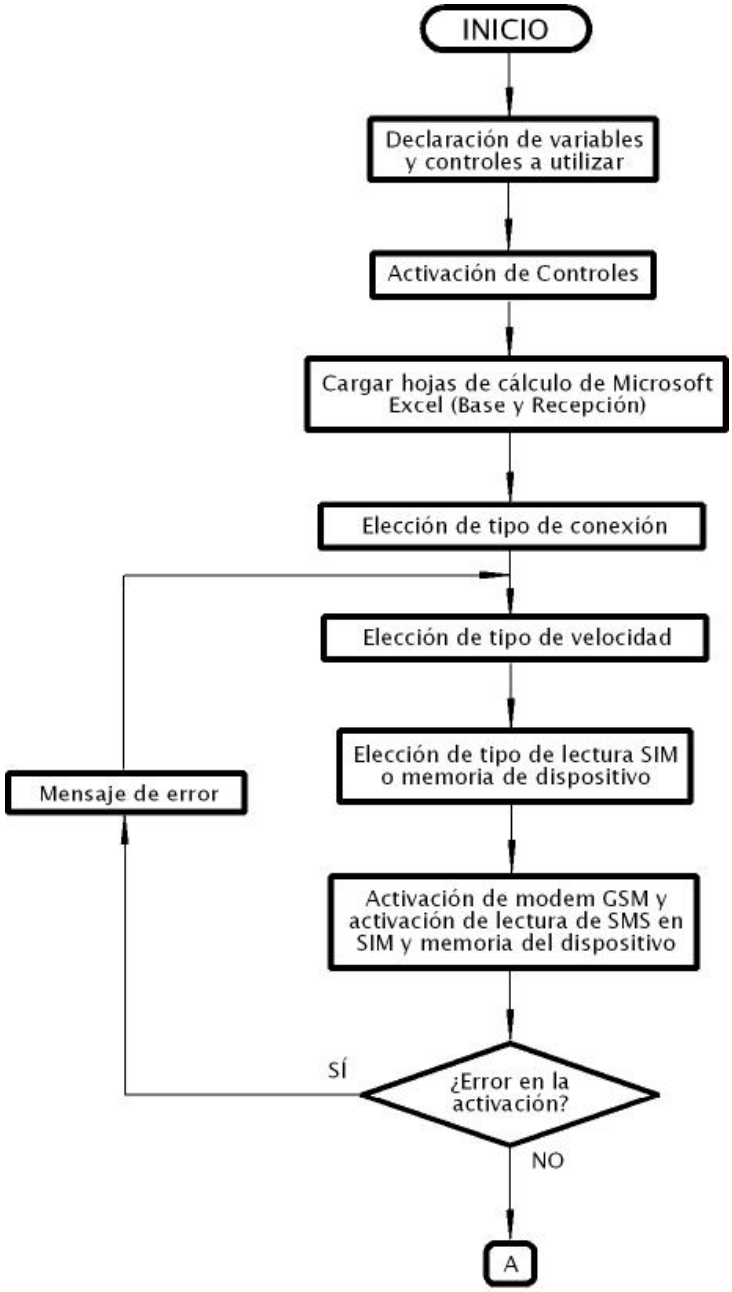
- Leer los mensajes de texto de la bandeja de entrada del teléfono.
- Comprobar si el SMS recibido es de un equipo móvil válido.
- Desempaquetar la información del SMS y ubicarla en los campos correspondientes en una tabla de Microsoft Excel.
- Gestionar y asegurar el envío un SMS de acuse de recibido hacia el equipo móvil que origino la comunicación.
- Manipular el control y la vida de la tabla de registros creada.

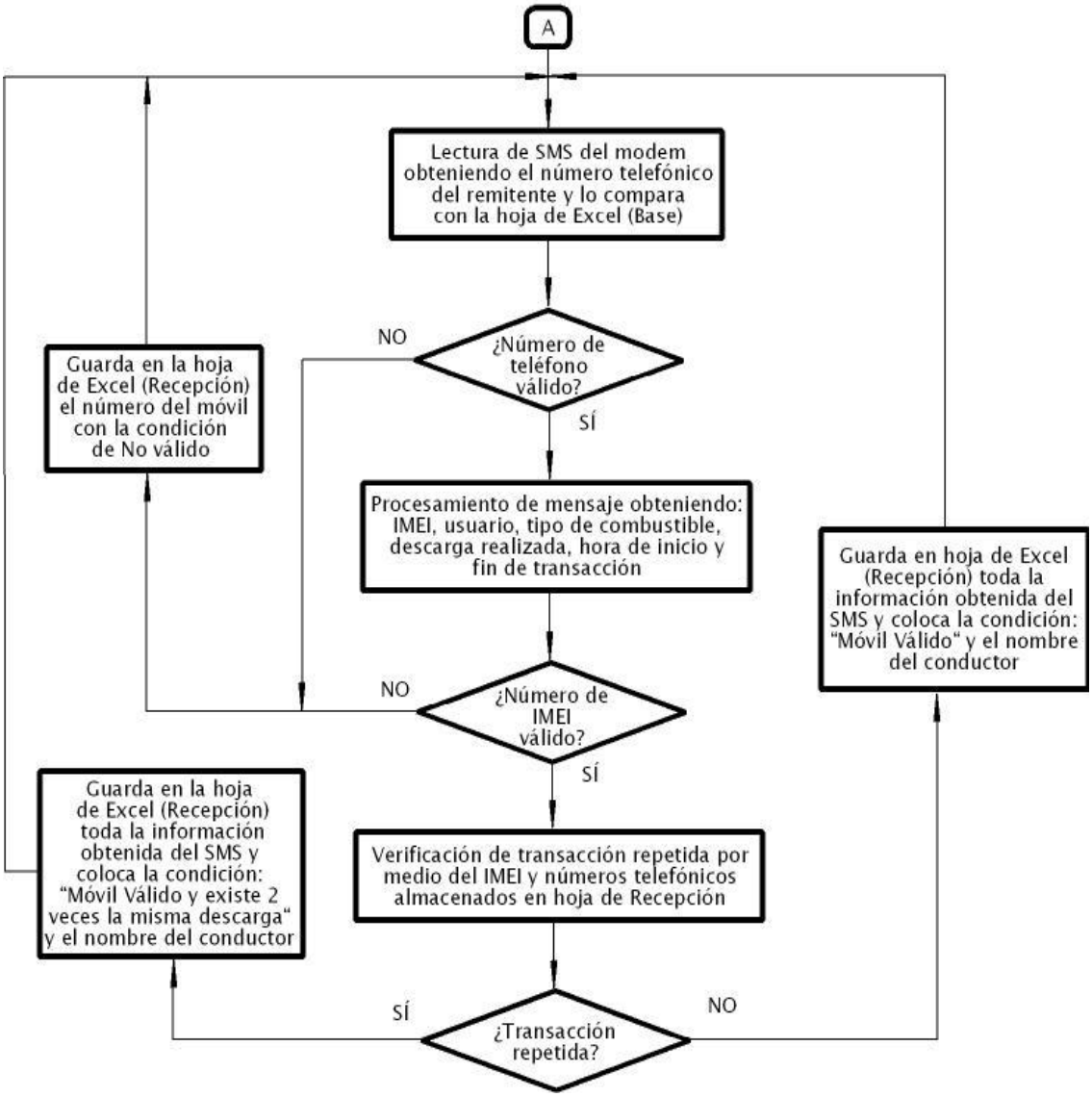
El software desarrollado que efectúa las tareas descritas anteriormente, se basa en el algoritmo que se describe a continuación.

- Presentar la pantalla el flash de inicio de programa.
- Ocultar flash y desplegar pantalla principal del programa.
- En la pantalla principal seleccionar el dispositivo de conexión (Modem) a utilizar.
- Elegir el tipo de lectura a realizar al Modem GSM, SIM o memoria del dispositivo.
- Seleccionar la velocidad de conexión del dispositivo.
- Activar la lectura de los mensajes de texto (SMS) recibidos en el Modem GSM.

- Verificar si el móvil que envió el SMS obtenido del Modem pertenece a la base de datos de los móviles validos, comprobando el número telefónico y el IMEI del mismo.
- Si no existe problema se obtiene del SMS los datos de: usuario, estación de servicio, hora de inicio de transacción, hora de finalización de transacción, tipo de combustible y descarga realizada guardándose estos datos en variables en la memoria; de no pertenecer a la base de datos no se realiza ningún proceso.
- Luego guardar los datos obtenidos del móvil en la base de datos para llevar un registro de los mismos.
- Enviar un mensaje de texto al móvil válido para indicar que el proceso de actualización de la base de datos ha concluido con éxito.
- Al finalizar el día se guarda la base de datos. El nombre de cada base de datos corresponderá a la fecha de cada día finalizado.

5.4.2. FLUJOGRAMA.




A
Lectura de SMS del modem obteniendo el número telefónico del remitente y lo compara con la hoja de Excel (Base)
¿Número de teléfono válido?
NO
SÍ
Guarda en la hoja de Excel (Recepción) el número del móvil con la condición de No válido
Procesamiento de mensaje obteniendo: IMEI, usuario, tipo de combustible, descarga realizada, hora de inicio y fin de transacción
¿Número de IMEI válido?
NO
SÍ
Guarda en hoja de Excel (Recepción) toda la información obtenida del SMS y coloca la condición: "Móvil Válido" y el nombre del conductor
Verificación de transacción repetida por medio del IMEI y números telefónicos almacenados en hoja de Recepción
¿Transacción repetida?
SÍ
NO
Guarda en la hoja de Excel (Recepción) toda la información obtenida del SMS y coloca la condición: "Móvil Válido y existe 2 veces la misma descarga" y el nombre del conductor

CAPÍTULO VI: *GUÍA DE USO.*

6.1. CARACTERÍSTICAS DEL HARDWARE.

El hardware desarrollado se implementa en tres tarjetas de circuito impreso, que junto con los dispositivos externos necesarios para el equipo móvil, se alojan en un armario compacto para una manipulación sencilla. A continuación se separan las características relacionadas al hardware en tres secciones diferentes con e objetivo de presentar un panorama más claro.

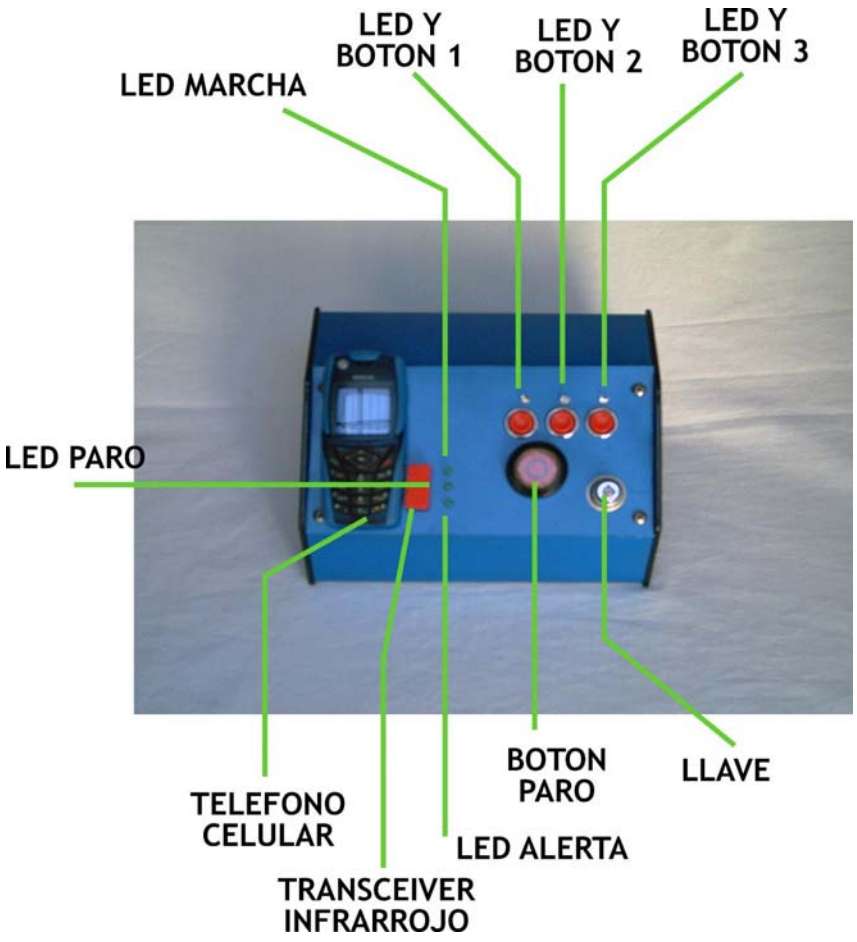
6.1.1. CARACTERÍSTICAS PERCEPTIBLES POR EL USUARIO.

Desde el punto de vista del usuario, el prototipo tiene las características descritas en la siguiente tabla:

Dispositivo	Función
Teléfono GSM	Inicio y cierre del sistema, transferencia de datos hacia la estación central
Llave de seguridad	Inicio y finalización de una transacción.
Botón 1	Activar despacho de combustible tipo A.
Botón 2	Activar despacho de combustible tipo B.
Botón 3	Activar despacho de combustible tipo C.
Botón de Paro	Finalizar del despacho de cualquier tipo de combustible.
Led 1 (Azul)	Indicar que se está despachando combustible tipo A.
Led 2 (Naranja)	Indicar que se está despachando combustible tipo B.
Led 3 (Rojo)	Indicar que se está despachando combustible tipo C.
Led 4 (Verde)	Indicador del estado de Marcha.
Led 5 (Verde)	Indicador del estado de Paro.
Led 6 (Verde)	Indicador del estado de Alerta.
Terminal Block	Conexiones eléctricas para la alimentación y dispositivos externos.
Portafusible	Protección contra cortocircuitos para el sistema de toma de alimentación

Tabla 6.1 Características visibles al usuario del prototipo construido.

Las imágenes siguientes presentan de forma pictórica los paneles frontal y posterior del armario construido, que aloja al sistema electrónico móvil, encargado de la adquisición, procesamiento y envío de los datos que reflejan los volúmenes descargados, hacia la estación central.



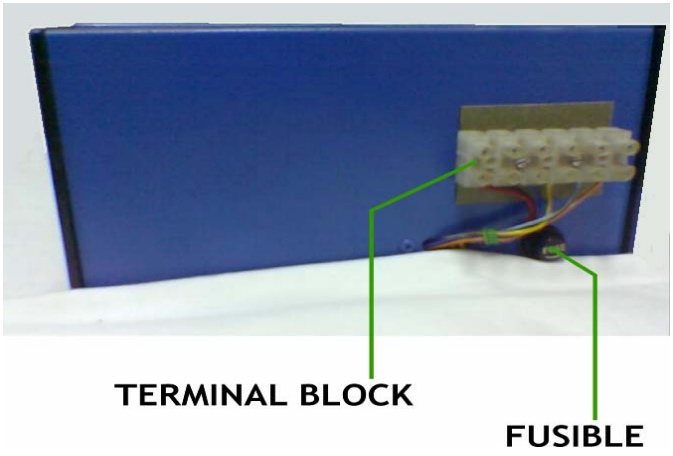


Figura 6.1 Fotografías del panel frontal y posterior gabinete construido.

6.1.2. CARACTERÍSTICAS MECÁNICAS.

Las características mecánicas más importantes que se reflejan del prototipo construido son:

- Rango de flujo medible: 0.04 lts./min. ~ 1.5 lts./min.
- Tuberías de 5mm Ø
- Armario de metal - plástico con medidas 20 cms. x 14 cms. x 11 cms.

Es oportuno recordar que las características mecánicas sobre el manejo del fluido, se relacionan directamente con las capacidades del flujómetro didáctico de Lucas Nülle.

6.1.3. CARACTERÍSTICAS ELÉCTRICAS.

Dentro de todas las características eléctricas inmersas en el proyecto, se enlistan a continuación las más destacables:

- Sistema gobernado por Microcontroladores PIC16F877A.
- 3 canales analógicos de 0V a +15V.
- 3 salidas de potencia de +12V y 500 mA a colector abierto.
- Interfaz IrDA a 56000 bps para teléfonos Nokia.
- Alimentado con fuente bipolar de +12V y -9V.
- Comunicación con el usuario a través de dispositivos electromecánicos y led's.
- Sistema distribuido en tres tarjetas de circuito impreso: Tarjeta I/O, Tarjeta MCU y Tarjeta de Comunicación IrDA.
- Conexiones eléctricas realizadas con conectores de presión o bloques terminales para facilitar el mantenimiento.
- Protección contra conexiones inversas en la toma de alimentación y canales analógicos.
- Protección contra sobretensiones en las salidas de potencia.

- Protección contra cortocircuitos para el sistema eléctrico de donde se tome la alimentación.

6.2. CARACTERÍSTICAS DEL SOFTWARE.

Ya que el software se encuentra distribuido en dos puntos y sobre dispositivos con características completamente diferentes, se realiza una separación para describir las características de cara al usuario que utilizará el dispositivo correspondiente.

6.2.1. EQUIPO MÓVIL.

Tal como se menciono en los capítulos anteriores, el MIDlet Hermes se ejecuta en un teléfono GSM Nokia 5140 y cuenta con las siguientes características:

- Desarrollado en J2ME bajo CLDC 1.1 / MIDP 2.0.
- Presentación de ayuda al iniciar la ejecución del MIDlet.
- Presentación de elementos gráficos para indicar el estado o fase operativa que se está ejecutando en un momento determinado.
- Uso de text box para ingresar la información.
- Ingreso de caracteres alfabéticos en formato minúscula/mayúscula, números y símbolos especiales soportados por el teléfono.
- Uso de las teclas de menú y de navegación para la interacción usuario-software.
- Parámetros de comunicación IrDA predeterminados y sin opción a cambios por parte del usuario.

6.2.2. ESTACIÓN CENTRAL.

Por su parte el software residente en la estación central se ejecuta en una computadora de escritorio con muchos más recursos computacionales que el Nokia 5140, y presenta las siguientes características:

- Desarrollado bajo Microsoft Visual Basic 6.0
- Entorno visual enriquecido y de fácil operación.
- Opción para cambiar por parte del usuario, los parámetros de conexión con el teléfono móvil dedicado.
- Almacenamiento de las transacciones por día, o cada vez que el usuario considere oportuno por si es necesario realizar varios cortes administrativos en un mismo día.
- Visualización de la tabla de datos actual o de una previamente almacenada.

6.3. ¿CÓMO UTILIZAR EL PROTOTIPO CONSTRUIDO?

Con el objetivo de presentar los pasos necesarios para la operación del prototipo, al igual que con la descripción del software, éstos se describen en dos etapas diferentes debido a la separación geografía existente entre las dos aplicaciones involucradas en la arquitectura del proyecto.

6.3.1. EQUIPO MÓVIL.

Los pasos necesarios para operar el equipo móvil, con el objetivo de realizar una transacción, se enlistan a continuación:

1. Entrar al menú del Nokia 5140.
2. Teclear la secuencia numérica **9-2-1**, para llegar hasta la carpeta donde esta almacenado el MIDlet.
3. Con las teclas de navegación buscar la aplicación GasComm y ejecutarla presionando la tecla **Eligir**.

4. Ejecutar el MIDlet Hermes presionando la tecla **Elegir**.
5. Al iniciar la ejecución de Hermes, se presenta una pantalla con la ayuda para facilitar el uso del mismo. Luego de leer la información, presionar **Aceptar**.

Figura 6.2 Pantalla de Información para el usuario.

6. Luego se presenta la pantalla para ingresar el código de acceso. Después de ingresarlo, presionar **Ok**.

Figura 6.3 Pantalla para ingresar el código de acceso a la aplicación.

7. Si el código es correcto, aparecerá la pantalla para ingresar el código usuario. Al concluir la escritura, presionar **Ok**.
8. De igual forma si el código es correcto, se presentará una pantalla para ingresar el código de la estación en donde se realizará la transacción. Luego de ingresarlo, presionar **Ok**.

Figura 6.4 Pantallas para ingresar los códigos de estacion y usuario.

9. Girar la llave de seguridad para encender el sistema electrónico y de esta forma iniciar el despacho de combustible. El indicador de marcha se mantendrá encendido, luego de esta acción.
10. Presionar el pulsador para despachar un tipo de combustible en particular. Se mantendrá activo un indicador del tipo de combustible seleccionado mientras se está en el proceso de descarga.
11. Para cancelar la carga de combustible se debe presionar el botón de paro y mantenerlo hasta que el indicador de paro se active.
12. Si se desea descargar otro tipo de combustible, se debe repetir los pasos 10 y 11 según sea necesario.
13. Cuando se ha terminado de descargar el combustible se debe cerrar la llave y con eso se da por finalizada la transacción en esa estación de servicio.
14. Automáticamente el teléfono genera un tono de aviso solicitando la confirmación para enviar un mensaje de texto, se debe presionar Si.
15. La aplicación regresara a la pantalla de ayuda descrita en el paso 5. Si la jornada no ha finalizado se deberá iniciar la nueva transacción desde éste paso. Si ya no se realizaran nuevas transacciones, se debe presionar la opción **Quitar**.

6.3.2. SOFTWARE DE LA ESTACIÓN CENTRAL.

1. Ejecutar el programa “Base de Datos Beta 1.001” desde el escritorio de Windows. Al ejecutarse mostrara la siguiente ventana la cual durara unos pocos segundos.

Figura 6.5 Pantalla de presentación del software administrativo.

2. Luego se abrirá la ventana principal. Para poner a funcionar el programa correctamente deben escogerse los parámetros correctos para gestionar la conexión con el modem.

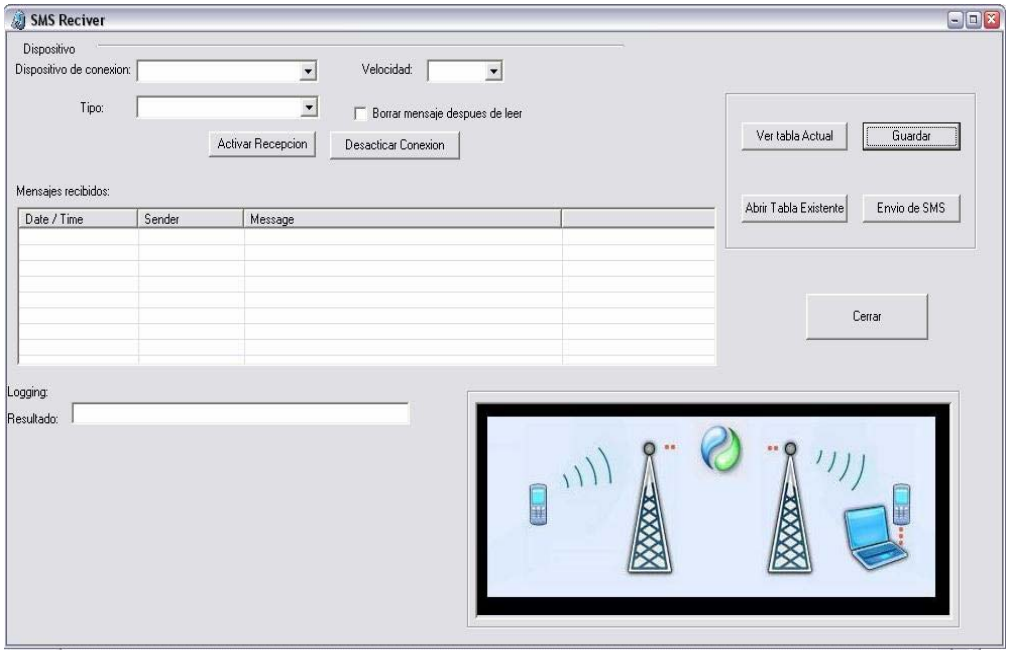


Figura 6.6 Pantalla de trabajo. Software Administrativo.

3. Seleccionar ***Siemens Serial Modem*** del la menú desplegable Dispositivo de Conexión.

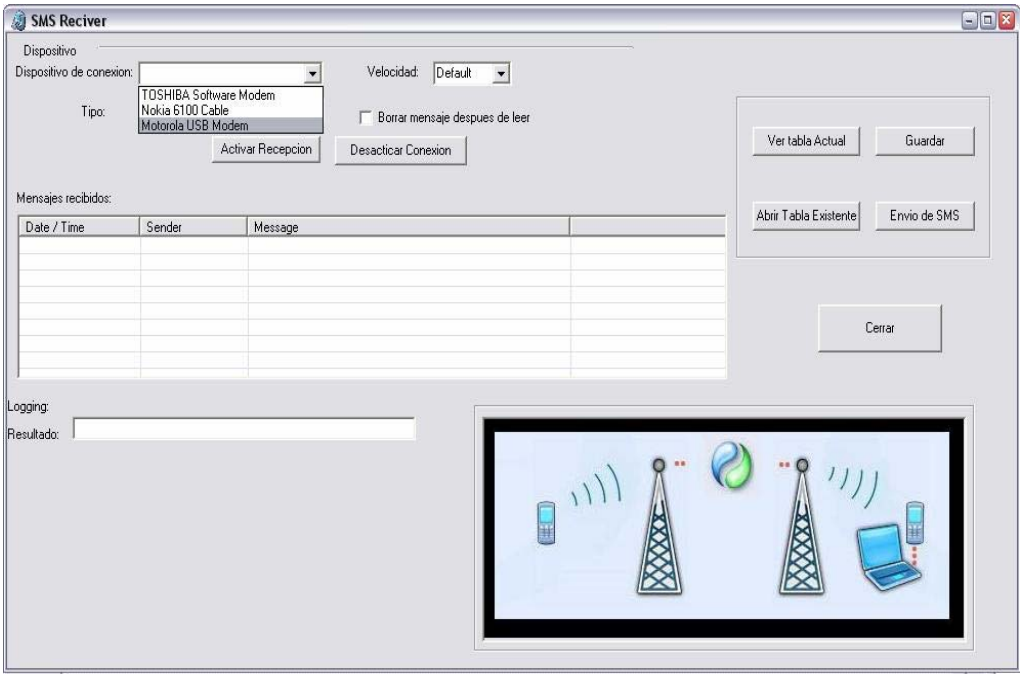


Figura 6.7 Selección del modem.

4. Elegir ***MT – SIM & Device Memory*** del menú desplegable Tipo.

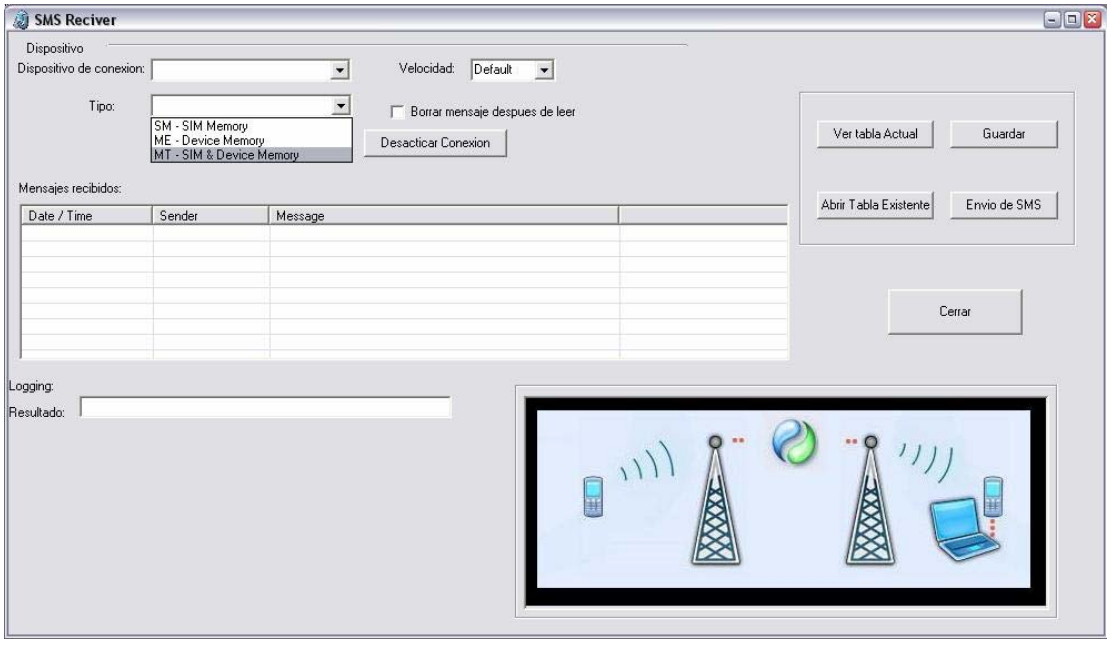


Figura 6.8 Selección de tipos de memoria a leer. SIM y Memoria del Teléfono.

5. Por último, seleccionar ***Default*** del menú desplegable Velocidad.

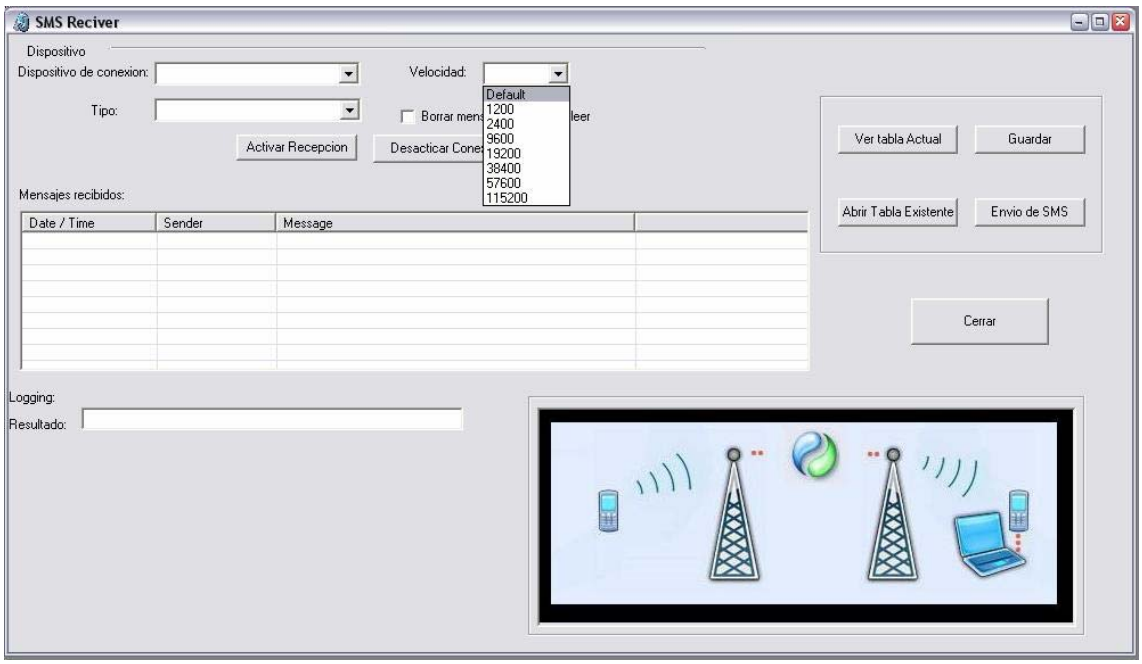


Figura 6.9 Selección de velocidad de conexión.

6. Después de configurar la conexión del dispositivo se activa la lectura del mismo por medio del botón ***Activar Recepción.***
7. Para detener la lectura del dispositivo de conexión se hace por medio del botón ***Desactivar Conexión*** el cual detendrá el proceso.
8. El programa posee la opción de ver tabla Actual la cual se activa por medio del botón ***Ver tabla Actual***, con esta opción se puede observar la tabla que se esta llenando en ese momento.
9. Si desea guardar la tabla actual lo puede hacer por medio del botón ***Guardar*** el cual realizara el proceso almacenar la tabla en la PC colocándole el nombre de la fecha del día que lo ha realizado y la condición o veces que lo ha realizado Ej.: 16-Ago-05[1].
10. Base de Datos Beta 1.001 ofrece la opción de observar tablas ya existentes en la PC, por medio del botón ***Abrir Tablas Existentes.***

CONCLUSIONES

- Pese a su número reducido de instrucciones, escasas variables de uso general (solamente una) y memoria de capacidad un tanto limitada, los microcontroladores PIC ofrece gran versatilidad en cuanto a lo que se refiere a costo de adquisición, requerimientos simples de hardware para programación e implementación, software accesible para simulación y un rendimiento tan alto como muchos otros microcontroladores de mayor costo y capacidad.

- Dados los requerimientos del proyecto, en lo que respecta a cantidad de memoria y recursos de hardware (puertos digitales, puertos analógicos, puertos de comunicación seriales y temporizadores), se eligió el microcontrolador de gama media PIC 16F877A ya que este satisface las necesidades la aplicación.

- Por la orientación del proyecto, en lo que respecta al desarrollo y diseño del software las opciones mas apropiadas utilizadas fueron Nokia PC Suite 6.70.22, Java Micro Edition (J2ME), NetBeans 5.0 y Microsoft Visual Basic 6.0; además de herramientas gratuitas encontradas en Internet para Visual Basic.

- En el proceso de investigación para determinar el móvil idóneo que soportara tanto el hardware como el software de la aplicación, se encontró que éste debe poseer las siguientes características: un perfil móvil MIDP 2.0, una configuración de CLDC 1.1 y los API Wireless Messaging, Mobile Media y Nokia UI.

- Para lograr establecer la comunicación entre el equipo móvil y la estación central, se desarrollo una pequeña aplicación en J2ME para determinar el número de puerto SMS válido, debido a que esta información es restringida por parte del operador y el fabricante.

- El estándar de comunicación serial RS232, a pesar de ser una interfaz de comunicación abierta, no es posible utilizarla como tal cuando se desea implementar en los teléfonos

Nokia. Esto es debido a que, hasta la fecha, el fabricante no permite un manejo abierto del puerto serial (com) de sus teléfonos móviles mediante aplicaciones desarrolladas en JavaME, específicamente en el envío/recepción de datos vía seriales.

- El estándar de comunicación infrarroja IrDA, al igual que la interfaz serie, no es posible utilizarla como tal hasta la fecha cuando se desea implementar en los teléfonos Nokia. Esto es debido a que ellos llevan asociada una codificación o encriptamiento en la comunicación que restringe su uso únicamente entre dispositivos Nokia, y no entre cualquier otro dispositivo o hardware que disponga también de una interfaz IrDA estándar. Asimismo se concluye que, hasta la fecha, el fabricante no permite un manejo abierto del puerto infrarrojo de sus teléfonos móviles mediante aplicaciones desarrolladas en JavaME, específicamente en el envío/recepción de datos vía infrarroja.

- En el desarrollo de Base de Datos Beta 1.001 se estableció la comunicación entre la PC y el modem GSM mediante Microsoft Visual Basic 6.0 y controles Active X; la limitante que se tubo para el manejo de los datos que se recibían del modem GSM a una base de datos de gran potencia como por ejemplo Microsoft Access es que, el control Active X seleccionado provoca conflictos en el manejo que posee Microsoft Visual Basic de Bases de Datos.

- Para solucionar este problema se auxilio de otra herramienta que posee Microsoft Visual Basic y es el manejo de tabla en Microsoft Excel creándose por medio de estas tablas en donde se almacenan los datos obtenidos del modem GSM; estos datos o tablas son guardados en la PC.

BIBLIOGRAFÍA Y FUENTES CONSULTADAS

1. El proceso de la investigación. Mario Tamayo y Tamayo. Segunda edición. Editorial Limusa.

2. Página Web de la Superintendencia General de Electricidad y Telecomunicaciones: www.siget.gob.sv
   Última visita: 20 Marzo 2006

3. Página Web del Informe sobre el Desarrollo Humano en El Salvador: www.desarrollohumano.org.sv
   Última visita: 20 Marzo 2006

4. Página Web de la Bolsa de Valores de El Salvador: www.bves.com.sv
   Última visita: 20 Marzo 2006

5. Página Web de Telefónica de El Salvador: www.telefonica.com.sv/movistar/nempresa/conocernos1.html
   Última visita: Mayo 2005

6. Apuntes de clases de la Cátedra de Sistemas de Comunicación Móvil. Universidad Don Bosco 2003.

7. Página Web del grupo GSM http://www.gsmworld.com/index.shtml
   Última visita: 20 Marzo 2006

8. Página Web de The 3rd Generation Partnership Project http://www.3gpp.org/
   Última visita: 20 Marzo 2006

9. Información escrita proporcionada por el señor Subdirector de Hidrocarburos y Minas del Ministerio de Economía de El Salvador, Lic. Francisco Cruz.

10. Instrumentación. Transductores e interfaz. B.R. Bannister y D.G. Whitehead. Segunda edición 1994. Addison-Wesley Iberoamericana.

11. Enciclopedia de la electrónica. Ingeniería y Técnica. C. Belove 1990 Editorial OCEANO/CENTRUM.

12. Sensores Acondicionadores y Procesadores de señal. Jordi Mayné. 2003. Página Web: http://www.ct.upc.es/departaments/eel/JCEE/JCEE2002/MAYNEPONENCIA.pdf
    Última visita: 20 Marzo 2006

13. Apuntes de clases de la Cátedra de Instrumentación Industrial.
Universidad Don Bosco 2004.

14. Microcontroladores PIC. Diseño práctico de aplicaciones. José María Angulo Usategui e Ignacio Angulo Martínez. Segunda edición 1999. McGraw-Hill Interamericana.

15. Comunicaciones y redes de computadoras. W. Stallings. Quinta Edicion 1997. Prentice Hall.

16. Serial Infrared Physical Layer Specification. Infrared Data Association. Version 1.3 2001. Página Web: www.irda.org
Última visita: 20 Marzo 2006

17. Universal Serial Bus Specification. Compaq / Intel / Microsoft / NEC Revision 1.1 1998. Pagina Web: www.usb.org
Última visita: 20 Marzo 2006

18. Universal Serial Bus Specification. Compaq / Hewlett-Packard / Intel / Lucent / Microsoft NEC / Philips. Revision 2.0 2000. Página Web: www.usb.org
Última visita: 20 Marzo 2006

19. Telecomunicaciones móviles. Eugenio Rey. Primera edición 1995. Alfaomega Marcombo.

20. Estudio comparativo de las técnicas de acceso múltiple utilizadas en telefonía celular. Wenceslao Rivas. Científica Año 1, número 2. Diciembre 2000. Universidad Don Bosco

21. Evolución de los sistemas móviles celulares GSM. Álvaro Pachón de la Cruz. 2004. Página Web:
http://www.icesi.edu.co/es/publicaciones/publicaciones/contenidos/sistemas_telematica/4/apachon_gsm.pdf
Última visita: 20 Marzo 2006

22. Cellular Wireless Modem Primer. 2004. Multi-Tech Systems, Inc. Página Web: http://www.multitech.com/DOCUMENTS/Tutorials/primers/
Última visita: 20 Marzo 2006

23. Manual de referencia Java 2. Herbert Schildt. Cuarta Edición. 2001. Osborne McGraw-Hill.

24. Java a tope: J2ME. Sergio Gálvez Rojas y Lucas Ortega Díaz. 2003. Universidad de Málaga. Página Web: http://www.lce.uma.es/~galvez/ftp/libros/J2ME.pdf
Última visita: Octubre 2004

25. Comunicaciones y bases de datos con Java a través de ejemplos. Adela Sancho y Jesús Bobadilla. Primera Edición 2003. Editorial Ra-Ma.

26. Aprendiendo Microsoft Access 2000 en 24 horas. Craig Eddy y Timothy Buchanan. Primera edición. 1999. Prentice Hall Hispanoamericana.

27. Manual del Instructor de Sistemas de Medición Volumétrica de Lucas Nülle. Laboratorio de Instrumentación y Control, Edificio de Electrónica, CITT

28. WT-S Stainless-Steel Turbine Meter Instructions. SeaMetrics. Página Web: http://www.seametrics.com
Última visita: 20 Marzo 2006

29. PIC16F87xA Data Sheet. Microchip 2003. Página Web: www.microchip.com
Última visita: 20 Marzo 2006

30. Mid-Range MCU Family Reference Manual. Microchip 1997. Página Web: www.microchip.com
Última visita: 20 Marzo 2006

31. Forum Nokia – Developer Resources. Pagina Web: http://www.forum.nokia.com
Última visita: 20 Marzo 2006

APÉNDICE A. PROCEDIMIENTO DE CONTRUCCIÓN DEL PROTOTIPO.

- ASPECTOS TÉCNICOS INICIALES CONSIDERADOS EN EL DISEÑO DEL PROTOTIPO.

Durante el periodo de definición conceptual del proyecto, se precisaron cuatro bloques fundamentales junto con la delimitación de las características técnicas que cada uno conllevaría, las cuales se concentran en la siguiente tabla:

Ubicación	Bloque	Funciones
Equipo móvil	Sistema electrónico de adquisición (gobernado por microncotrolador)	• Interactuar con el usuario a través de dispositivos de mando y señalización. • Obtener, acondicionar y digitalizar la señal eléctrica proveniente del sensor de flujo. • Procesar la información obtenida de caudal, a fin de obtener el volumen descargado, para su posterior transmisión. • Comunicarse con el teléfono GSM (del equipo móvil) a partir de una interfaz infraroja que cumpla el estándar IrDA.
	Teléfono GSM (midlet java)	• Habilitar el inicio de una transacción. (via IrDA) • Obtener los datos de volumenes descargados. • Empaquetar la información relativa a la transacción en un SMS.

Ubicación	Bloque	Funciones
Estación Central	Telefono GSM (gobernado por PC estación central).	• Recibir los mensajes de las transacciones realizadas y enviar sms de acuse de recibido.
	Computadora Central (software administrativo)	• Descargar los SMS de las transacciones realizadas desde la bandeja de entrada del teléfono con el objetivo de desempaquetar la información útil. • Desglosar la información contenida en el sms y formar una tabla para su presentación al usuario. • Administrar el almacenamiento y disponibilidad de las tablas de información para su uso. • Gestionar el envìo de SMS de acuse de recibido.

Tabla A1. Resumen de distribución de funciones del proyecto.

Con el panorama que se describió anteriormente, surgió la arquitectura del proyecto, que se muestra en la figura A2:

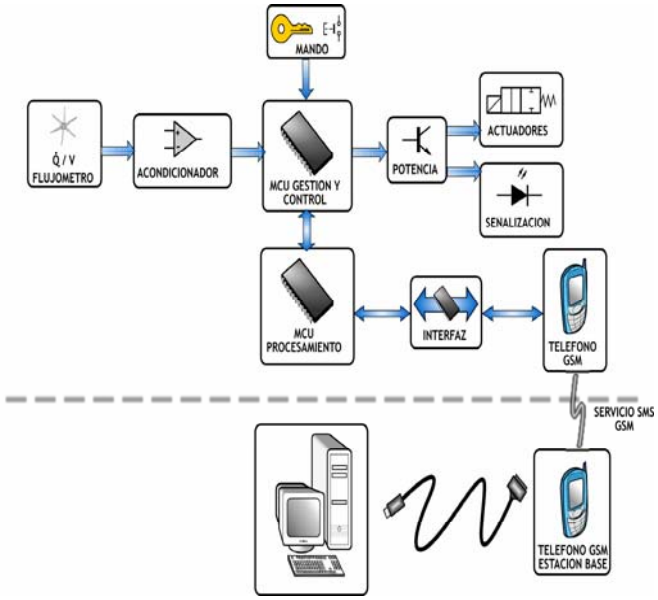


Figura A2. Arquitectura de la aplicación.

- OBSTÁCULOS TÉCNICOS ENCONTRADOS DURANTE EL DESARROLLO DEL PROTOTIPO.

El principal problema que se encontró durante el desarrollo del proyecto, y que lastimosamente no se pudo solventar, evitándose así que se cumplieran algunos objetivos trazados inicialmente, fue el lograr comunicar el teléfono GSM del equipo móvil con el sistema electrónico de adquisición. La dificultad primaria radica en la escasa o nula información, en algunos casos, por parte de los fabricantes de teléfonos móviles sobre el tratamiento de la comunicación, tanto a nivel de hardware y protocolo de comunicación. Pareciendo que el estándar no siempre se aplica rigurosamente en los equipos y que cada fabricante genera su propia forma de comunicación.

Ante dicha situación, con el afán de cumplir los objetivos trazados, se inicio una búsqueda de alguna otra interfaz que se adaptara al esquema de la arquitectura inicial, y que fuera soportada por el teléfono móvil disponible. Amparados a la poca información encontrada en páginas web de autores originarios de Europa oriental o escandinavos, se logro comprender la interfaz M_{BUS}/F_{BUS} (básicamente es un tipo de comunicación serie Half Duplex/Full Duplex) con la que disponen algunos teléfonos Nokia. Se realizaron pruebas en dos modelos, el modelo 3100 (CLDC 1.0 y MIDP 1.1) del cual se contaba con un diagrama eléctrico del equipo así como del pinout del puerto de comunicación (Pop-Port), con este teléfono se logro establecer comunicación pero lastimosamente no era apto para poder ejecutar la aplicación Java desarrollada. Extrapolando el circuito de prueba hacia el Nokia 6230 (CLDC 1.1 y MIDP 2.0), idóneo para el midLET Java, no se obtuvieron los reultados esperados ya que al acoplarse galvánicamente ambos circuitos, el teléfono sufrió daños en el puerto de comunicación, finalmente en un foro colombiano, se descubrió que el dispositivo utiliza un cable de tipo DKU-5 el cual permite comunicación USB (tanto del lado del teléfono como del lado de la computadora).

Optando por el modo de comunicación menos traumático para la arquitectura del sistema desarrollada, se regreso a IrDA e intentar lograr la comunicación utilizando un escazo grupo de instrucciones para el modelo 5140 que Nokia tiene disponible en su

portal web [31], dichas instrucciones se utilizaron en un Nokia 5140 configurado a 57.6 kbps, 8 bits de datos, 1 bit de paridad (configuración obtenida luego de descifrar los comandos AT utilizados por una PDA para comunicarse con el teléfono). Aún así, utilizando las instrucciones propuestas por Nokia para el mismo modelo, no se logró establecer comunicación.

- CONTEXTO DE PRUEBA.

Dentro de este ambiente, se generó una reunión con todos los miembros involucrados en el proceso, y se tomo la decisión de documentar de forma explícita todo el trabajo realizado, con el objeto de que posteriormente se pueda retomar el tema y poder desarrollar de manera satisfactoria la apertura del puerto infrarrojo para establecer comunicación entre ambas partes. Considerando la recomendación se logro concluir el prototipo con las funciones que se describen en la tabla A3 y con una arquitectura como la presentada en la figura A4.

Ubicación	Bloque	Funciones
Equipo Móvil	Sistema Electrónico de Adquisición (gobernado por microncotrolador)	• Interactuar con el usuario a través de dispositivos de mando y señalización. • Obtener, acondicionar y digitalizar la señal eléctrica proveniente del sensor de flujo. • Procesar la información obtenida de caudal, a fin de obtener el volumen descargado, para su posterior transmisión. • Con motivo de poder visualizar la transferencia serie, se diseño una interfaz y una pequeña aplicación en visual basic
	Teléfono GSM (midlet java)	• Empaquetar la información relativa a la transacción en un SMS (a modo de simulación se dejo la opción de poder realizar ciertas modificaciones desde la aplicación java, para poder visualizar en la etapa receptora cierta diversidad de mensajes de texto).

Ubicación	Bloque	Funciones
Estación Central	Telefono GSM (gobernado por pc estación central).	• Recibir los mensajes de las transacciones realizadas y enviar SMS de acuse de recibido.
	Computadora Central (software administrativo)	• Descargar los SMS de las transacciones realizadas desde la bandeja de entrada del teléfono con el objetivo de desempaquetar la información útil. • Desglosar la información contenida en el sms y formar una tabla para su presentación al usuario. • Administrar el almacenamiento y disponibilidad de las tablas de información para su uso. • Gestionar el envìo de SMS de acuse de recibido.

Tabla A3. Resumen de funciones disminuidas en la entrega final del prototipo.

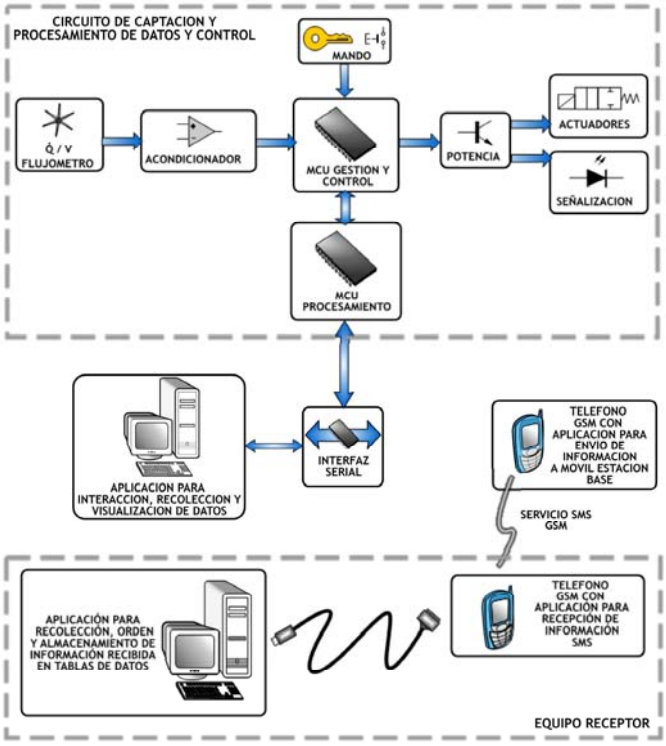


Figura A4. Arquitectura modificada del prototipo.

- PROGRAMA DEMOSTRATIVO EN VISUAL BASIC.

En vista de que no fue posible establecer una comunicación vía infrarroja entre el dispositivo móvil y el circuito, para efectos de demostración se elaboró el siguiente programa que permite simular dicho móvil manipulando el puerto serie RS-232 de la PC mediante Visual Basic 6.0.

Figura A5. Ambiente visual del programa.

Se trata de una interfaz gráfica que permite visualizar el volumen en litros que corresponde a cada electro válvula, además de poder enviar datos en respuesta al microcontrolador para completar la secuencia lógica de la comunicación.

Botón “Simular error”: Mediante este botón, se envían una secuencia de datos al microcontrolador tal que éste los interpreta como un error en la transmisión hacia el móvil. Como respuesta, el microcontrolador reenvía todos los datos de todas las electroválvulas.

Botón “Salir”: Finaliza la aplicación

Botón “Enviar un dato”: Por medio de este botón se envía un dato al microcontrolador que le permite ya sea habilitar el sistema al inicio de una operación, o bien reiniciarse al finalizar una operación exitosa.

Código del Programa.

```
Dim dato As String        'Variable utilizada para capturar el dato recibido en el puerto
                          'serie
Dim var1 As Integer       'Variable que guarda el número equivalente al código ASCII
                          'recibido en "dato"
Dim ev As Integer         'Variable que almacena el número de la electro válvula a la
                          'cual pertenecen ciertos datos
Dim cont As Integer       'Variable empleada para diferenciar el tipo de dato contenido
                          'en var1
Dim valor As Double       'Variable utilizada para el cálculo de litros que se mostrará
                          'en pantalla
```

```
Private Sub Command1_Click()              'Acciones correspondientes al presionar el
                                          'botón "Salir"
   If MSComm1.PortOpen = True Then
     MSComm1.PortOpen = False             'Se asegura cerrar el puerto serie antes de
   End If                                 'finalizar esta aplicación

   End                                    'Se finaliza esta aplicación

End Sub
```

```
Private Sub Command2_Click()          'Acciones correspondientes al presionar el botón
                                      ' "Enviar un dato"
   MSComm1.Output = "7"               'Se envía un dato al puerto serie

   Do                                                  'Mediante este lazo se asegura no poder
   dato = dato                                         'enviar ningún otro dato mientras la
   Loop Until MSComm1.OutBufferCount = 0   'transmisión en curso no termine
```

```
End Sub
_____________________________________________________________________________

Private Sub Command4_Click()        'Acciones correspondientes al presionar el botón
                                                ' "Simular un error"
   MSComm1.Output = "7"
   MSComm1.Output = "7"             'Se envían dos datos consecutivos al puerto serie
   Do                                                          'Mediante este lazo se asegura no poder
   dato = dato                                                 'enviar ningún otro dato mientras la
   Loop Until MSComm1.OutBufferCount = 0    'transmisión en curso no termine

End Sub
_____________________________________________________________________________

Private Sub Form_Load()             'Acciones que se efectuarán al cargar el formulario

   If MSComm1.PortOpen = False Then
      MSComm1.PortOpen = True                'Se asegura abrir el puerto serie al iniciar
   End If                                                      'esta aplicación

      cont = 0                      'Se inicializa la variable cont a cero
      Text1.Text = ""               'Se inicializan los visualizadores de volumen en vacío
      Text2.Text = ""
      Text3.Text = ""

End Sub
_____________________________________________________________________________

Private Sub Timer1_Timer()              'Acciones correspondientes al Timer1

If MSComm1.InBufferCount <> 0 Then      'Se verifica que haya un dato recibido en el
                                                            'puerto serie
   dato = MSComm1.Input                 'Si hay un dato recibido, este se almacena
                                                            'en "dato"

   Select Case dato                 'Se determina el valor numérico equivalente al dato
                                                'ASCII guardado en "dato"
                                    'El número equivalente hallado se almacena en "var1"
         Case "ÿ"
            var1 = 255
         Case "þ"
            var1 = 254
         Case "ý"
            var1 = 253
         Case "ü"
            var1 = 252
         Case "û"
            var1 = 251
         Case "ú"
            var1 = 250
         Case "ù"
            var1 = 249
         Case "ø"
```

```
    var1 = 248
Case "÷"
    var1 = 247
Case "ö"
    var1 = 246
Case "õ"
    var1 = 245
Case "ô"
    var1 = 244
Case "ó"
    var1 = 243
Case "ò"
    var1 = 242
Case "ñ"
    var1 = 241
Case "ð"
    var1 = 240
Case "ï"
    var1 = 239
Case "î"
    var1 = 238
Case "í"
    var1 = 237
Case "ì"
    var1 = 236
Case "ë"
    var1 = 235
Case "ê"
    var1 = 234
Case "é"
    var1 = 233
Case "è"
    var1 = 232
Case "ç"
    var1 = 231
Case "æ"
    var1 = 230
Case "å"
    var1 = 229
Case "ä"
    var1 = 228
Case "ã"
    var1 = 227
Case "â"
    var1 = 226
Case "á"
    var1 = 225
Case "à"
    var1 = 224
Case "ß"
    var1 = 223
Case "Þ"
    var1 = 222
Case "Ý"
    var1 = 221
Case "Ü"
    var1 = 220
```

```
Case "Û"
   var1 = 219
Case "Ú"
   var1 = 218
Case "Ù"
   var1 = 217
Case "Ø"
   var1 = 216
Case "×"
   var1 = 215
Case "Ö"
   var1 = 214
Case "Õ"
   var1 = 213
Case "Ô"
   var1 = 212
Case "Ó"
   var1 = 211
Case "Ò"
   var1 = 210
Case "Ñ"
   var1 = 209
Case "Ð"
   var1 = 208
Case "Ï"
   var1 = 207
Case "Î"
   var1 = 206
Case "Í"
   var1 = 205
Case "Ì"
   var1 = 204
Case "Ë"
   var1 = 203
Case "Ê"
   var1 = 202
Case "É"
   var1 = 201
Case "È"
   var1 = 200
Case "Ç"
   var1 = 199
Case "Æ"
   var1 = 198
Case "Å"
   var1 = 197
Case "Ä"
   var1 = 196
Case "Ã"
   var1 = 195
Case "Â"
   var1 = 194
Case "Á"
   var1 = 193
Case "À"
   var1 = 192
Case "¿"
```

```
    var1 = 191
Case "¾"
    var1 = 190
Case "½"
    var1 = 189
Case "¼"
    var1 = 188
Case "»"
    var1 = 187
Case "º"
    var1 = 186
Case "¹"
    var1 = 185
Case "¸"
    var1 = 184
Case "·"
    var1 = 183
Case "¶"
    var1 = 182
Case "µ"
    var1 = 181
Case "´"
    var1 = 180
Case "³"
    var1 = 179
Case "²"
    var1 = 178
Case "±"
    var1 = 177
Case "°"
    var1 = 176
Case "¯"
    var1 = 175
Case "®"
    var1 = 174
Case "-"
    var1 = 173
Case "¬"
    var1 = 172
Case "«"
    var1 = 171
Case "ª"
    var1 = 170
Case "©"
    var1 = 169
Case "¨"
    var1 = 168
Case "§"
    var1 = 167
Case "¦"
    var1 = 165
Case "¥"
    var1 = 164
Case "¤"
    var1 = 163
Case "£"
    var1 = 162
```

```
Case "¢"
   var1 = 161
Case "¡"
   var1 = 160
Case " "
   var1 = 159
Case "Ÿ"
   var1 = 158
Case "ž"
   var1 = 157
Case "    "
   var1 = 156
Case "œ"
   var1 = 155
Case "›"
   var1 = 154
Case "š"
   var1 = 153
Case "™"
   var1 = 152
Case "˜"
   var1 = 151
Case "—"
   var1 = 150
Case "–"
   var1 = 149
Case "•"
   var1 = 148
Case """
   var1 = 147
Case "“"
   var1 = 146
Case "'"
   var1 = 145
Case "‘"
   var1 = 144
Case "    "
   var1 = 143
Case "    "
   var1 = 142
Case "Ž"
   var1 = 141
Case "    "
   var1 = 140
Case "Œ"
   var1 = 139
Case "‹"
   var1 = 138
Case "Š"
   var1 = 137
Case "‰"
   var1 = 136
Case "ˆ"
   var1 = 135
Case "‡"
   var1 = 134
Case "†"
```

```
    var1 = 133
Case "…"
    var1 = 132
Case "„"
    var1 = 131
Case "ƒ"
    var1 = 130
Case ","
    var1 = 129
Case "    "
    var1 = 128
Case "€"
    var1 = 127
Case " "
    var1 = 126
Case "~"
    var1 = 125
Case "}"
    var1 = 124
Case "|"
    var1 = 123
Case "{"
    var1 = 122
Case "z"
    var1 = 121
Case "y"
    var1 = 120
Case "x"
    var1 = 119
Case "w"
    var1 = 118
Case "v"
    var1 = 117
Case "u"
    var1 = 116
Case "t"
    var1 = 115
Case "s"
    var1 = 114
Case "r"
    var1 = 113
Case "q"
    var1 = 112
Case "p"
    var1 = 111
Case "o"
    var1 = 110
Case "n"
    var1 = 109
Case "m"
    var1 = 108
Case "l"
    var1 = 107
Case "k"
    var1 = 106
Case "j"
    var1 = 105
```

```
Case "i"
   var1 = 104
Case "h"
   var1 = 103
Case "g"
   var1 = 102
Case "f"
   var1 = 101
Case "e"
   var1 = 100
Case "d"
   var1 = 99
Case "c"
   var1 = 98
Case "b"
   var1 = 97
Case "a"
   var1 = 96
Case "`"
   var1 = 95
Case "_"
   var1 = 94
Case "^"
   var1 = 93
Case "]"
   var1 = 92
Case "\"
   var1 = 91
Case "["
   var1 = 90
Case "Z"
   var1 = 89
Case "Y"
   var1 = 88
Case "X"
   var1 = 87
Case "W"
   var1 = 86
Case "V"
   var1 = 85
Case "U"
   var1 = 84
Case "T"
   var1 = 83
Case "S"
   var1 = 82
Case "R"
   var1 = 81
Case "Q"
   var1 = 80
Case "P"
   var1 = 79
Case "O"
   var1 = 78
Case "N"
   var1 = 77
Case "M"
```

```
    var1 = 76
Case "L"
    var1 = 75
Case "K"
    var1 = 74
Case "J"
    var1 = 73
Case "I"
    var1 = 72
Case "H"
    var1 = 71
Case "G"
    var1 = 70
Case "F"
    var1 = 69
Case "E"
    var1 = 68
Case "D"
    var1 = 67
Case "C"
    var1 = 66
Case "B"
    var1 = 65
Case "A"
    var1 = 64
Case "@"
    var1 = 63
Case "?"
    var1 = 62
Case ">"
    var1 = 61
Case "="
    var1 = 60
Case "<"
    var1 = 59
Case ";"
    var1 = 58
Case ":"
    var1 = 57
Case "9"
    var1 = 56
Case "8"
    var1 = 55
Case "7"
    var1 = 54
Case "6"
    var1 = 53
Case "5"
    var1 = 52
Case "4"
    var1 = 51
Case "3"
    var1 = 50
Case "2"
    var1 = 49
Case "1"
    var1 = 48
```

```
Case "0"
    var1 = 47
Case "/"
    var1 = 46
Case "."
    var1 = 45
Case "-"
    var1 = 44
Case ","
    var1 = 43
Case "+"
    var1 = 42
Case "*"
    var1 = 41
Case ")"
    var1 = 40
 Case "("
    var1 = 39
Case """
    var1 = 38
Case "&"
    var1 = 37
Case "%"
    var1 = 36
Case "$"
    var1 = 35
Case "#"
    var1 = 34
Case """"
    var1 = 33
Case "!"
    var1 = 32
Case " "
    var1 = 31
Case ""
    var1 = 30
Case "-"
    var1 = 29
Case " "
    var1 = 28
Case " "
    var1 = 27
Case " "
    var1 = 26
Case " "
    var1 = 25
Case " "
    var1 = 24
Case " "
    var1 = 23
Case " "
    var1 = 22
Case " "
    var1 = 21
Case " "
    var1 = 20
Case " "
```

```
        var1 = 19
    Case " "
        var1 = 18
    Case " "
        var1 = 17
    Case " "
        var1 = 16
    Case " "
        var1 = 15
    Case " "
        var1 = 14
    Case " "
        var1 = 13
     Case " "
        var1 = 12
    Case " "
        var1 = 11
    Case " "
        var1 = 8
    Case " "
        var1 = 7
    Case " "
        var1 = 6
    Case " "
        var1 = 5
    Case " "
        var1 = 4
    Case " "
        var1 = 3
    Case " "
        var1 = 2
    Case " "
        var1 = 1
    Case Else
        var1 = 0
End Select


Select Case cont                'Se determina el procesamiento de los datos recibidos en
                                        'el puerto serie, según su orden de llegada.
                                'Para cada electro válvula, el orden de transmisión de los
                                        'datos en el microcontrolador son: número de ev, parte
                                        'alta valor calculado, parte baja valor calculado.
  Case 0
    ev = var1                   'El primer dato recibido se trata del número de
                                        'electro válvula. Este se almacena en "ev".
    cont = 1                    'Se modifica "cont" para indicar que el primer dato ya
                                        'fué recibido
  Case 1
    valor = var1 * 0.1 * 256        'El segundo dato recibido se trata de la parte alta (8
                                            'bits más significativos) del valor calculado por el
                                            'microcontrolador. Se multiplica anticipadamente
                                            'por 0.1 para prevenir un error por desbordamiento
                                            'en el producto var1 por 256.
    cont = 2                        'Se modifica "cont" para indicar que el segundo
                                            'dato ya fué recibido.
  Case 2
```

```
        valor = (valor + (var1 * 0.1)) * (5 / 1024)
'El tercer dato recibido se trata de la parte baja (8 bits menos significativos) del valor 'calculado por el microcontrolador. La
fórmula para completar el cálculo del volumen 'descargado en base a los datos recibidos es:
'               volumen = (valor recibido) * (5/1024) * 0.1

'donde:        valor recibido = (valor parte alta * 256) + (valor parte baja).
'                 Las dimensiones de esta cantidad están en "valor resultado ADC por
'                 minuto"

'                 5/1024 = es el valor de resolución del ADC del microcontrolador
'                 (5 = 5 voltios; 1024 = 2^10). Al efectuar el producto "valor devuelto *
'                 5/1024" se obtienen las dimensiones "voltios por minuto".
'                 0.1 = es el factor de equivalencia "voltios-flujo" del sensor de flujo
'                 (1V =0.1 litros/minuto). Al efectuar el producto "valor devuelto * 5/1024 *
'               0.1, se obtienen las dimensiones "litros"

          If ev = 1 Then                    'Se determina a qué electroválvula corresponde el
                                            'volumen obtenido para mostrarlo en pantalla
        Text1.Text = valor
        ElseIf ev = 2 Then
        Text2.Text = valor
        ElseIf ev = 3 Then
        Text3.Text = valor
        End If
        cont = 0

   End Select

End If

End Sub
```

Desde la línea en la cual se identifica var1 =28 puede observarse que, aparentemente, es el mismo caracter “ ” que se repite hasta llegar a var = 1. Aunque a simple vista el símbolo es el mismo, Visual Basic es capaz de diferenciar todos estos símbolos en base al valor en ASCII que cada uno de ellos conserva, aunque compartan el mismo símbolo representativo.

APÉNDICE B. HOJA TÉCNICA DEL FLUJÓMETRO DE SEAMETRICS

SeaMetrics

WT-S Stainless-Steel Turbine Meter Instructions

General Information

This unique system of 2" to 8" turbine meters uses one moving part, a precise helical rotor. Rotation of the rotor is electronically detected and processed. High-quality jewel bearings provide long wear life in non-lubricating fluids. The entire rotor assembly can be easily taken out of the meter for field service, without removing the meter from the pipe.

WTS bodies are fabricated from stainless steel tubing. The turbine insert is machined from a stainless steel casting. Turbine rotor is Kynar (PVDF).

An electronic register can be mounted on the meter to display flow rate, total (resettable or non-resettable) and provide a programmable pulse output. The same unit, in a wall mount or panel mount housing, can be located up to 2,000 feet away. It is not necessary to have any processing electronics on the meter itself unless local reading is desired. Other electronics options (which can again be meter or remote mounted) are the AO55 blind 4-20 mA transmitter, the PD10 divider, and a battery-operated (FT415) rate/totalizer.

Specifications

Materials

Meter Body	T-304 Stainless, T-316 optional
Turbine Insert	CF8M Cast Stainless
Turbine Rotor	Kynar (PVDF)
Shafts	Zirconia ceramic
Bearings	Sapphire journal, ruby ball

Maximum Pressure 200 psi (14 bar)

Maximum Temperature 200° F (93° C)

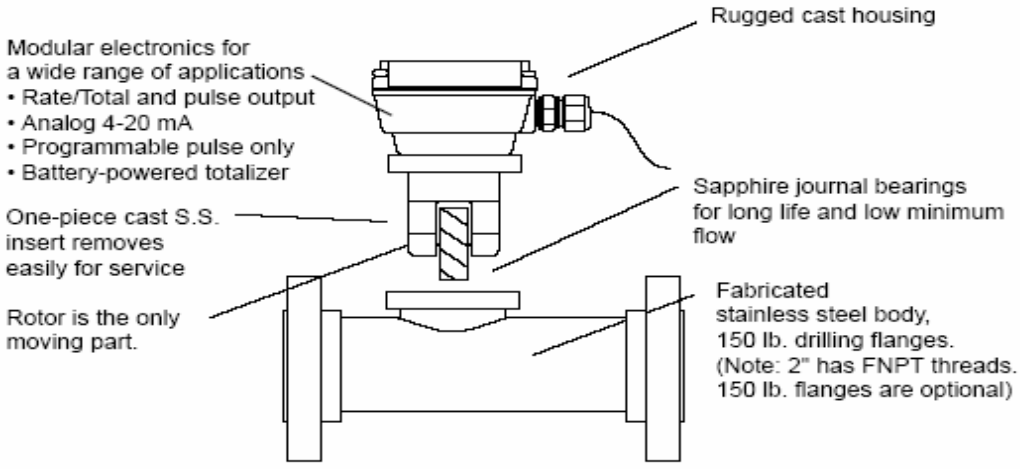
Accuracy ± 1% FS

Flow Range (GPM)

	2"	3"	4"	6"	8"
Min	2	3	6	12	30
Max	150	400	600	1200	3000

Cable #22 AWG 3-con, 18 ft. (6m)

Features

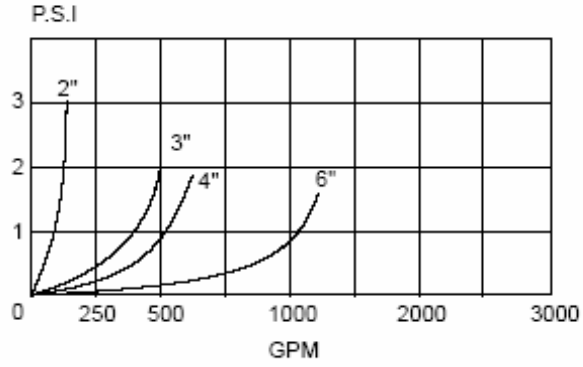


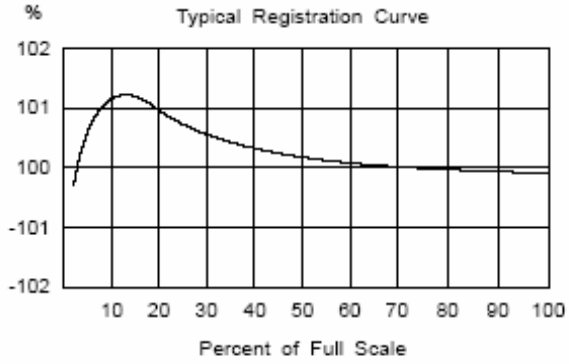
LT-10620-A
Page 1 Of 4

Electronic Options Specifications

WT100 (Pulse Only)	
Power	6-24 VDC
Pulse Type	Current sinking
WT101	
Power	12-32 VDC, 1.5 mA current loop powered 24 VDC optional
Rate	8-digit autorange
Total	8-digit, selectable decimal Reset standard, non-reset option
Memory	Nonvolatile (no battery needed)
K-factor Range	.050 - 1,999.999
Pulse Output	0.1 second, open collector
Pulse Range	0.1 - 99,999 gallons per pulse
Analog Output	4-20 mA, user-programmed span, two wire
WT102 (Blind Transmitter)	
Output	4-20 mA
Loop Power	12 - 36 VDC (isolated)
Accuracy	± 1%
Response Time	3 sec., 95% FS
WT104 (Battery-powered Ratemeter)	
Rate & Total	LCD readout w/resettable totalizer display
Battery Life	3 Years
Battery Type	Lithium, replaceable

Pressure Loss Chart



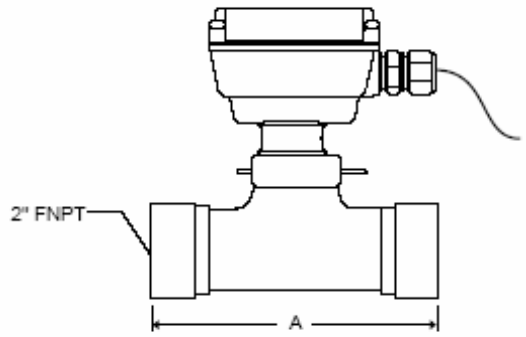


Dimensions

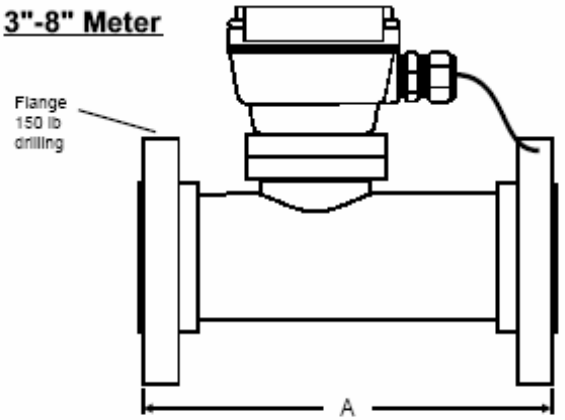
Meter size	Dimen. A
2"	8"*
3"	12"
4"	14"
6"	18"
8"	20"

* Female NPT threaded ends standard, flange or weld ends available

2" Meter

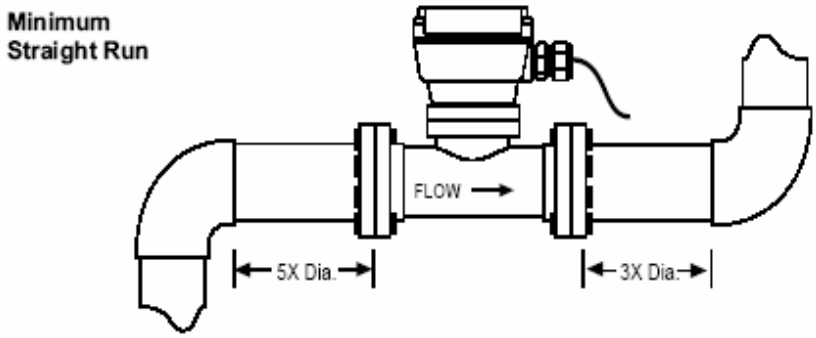


3"-8" Meter



Installation

These water meters are not recommended for installation downstream of the boiler feedwater pump where installation fault may expose the meter to boiler pressure and temperature. Maximum recommended temperature is 200°F.



Piping Conditions. In general, the standard practice of installing the meter with ten diameters of straight pipe upstream and five downstream are recommended. However, it is possible under some circumstances to operate with less, particularly if the meter is equipped with an optional internal flow straightener. (See diagram above.)

Flanges. Standard flanges are 150 lb. ANSI drilling. Either partial or full-face gaskets can be used. When installing, tighten the bolts evenly, and use care to prevent a misaligned gasket from entering the flow stream.

Position. The WT Series are all-position meters, and can be operated in a vertical or horizontal position, and with the meter insert in any radial position. A horizontal insert position is preferred if there is a risk of air becoming trapped due to constant low flows. Operating the meter in partially-filled pipe will result in inaccuracies.

Connections. Most WT meters require electrical connections. See the connections diagram for the one relevant to your meter.

Operation

For operating instructions for the various electronic modules, consult the manual for the specific module. This should be included with the meter when purchased.

Maintenance and Repair

Recalibration. If it is necessary to recalibrate the meter for any purpose, this can be done by any SeaMetrics-authorized facility. Call the factory for information.

Turbine Insert Removal and Installation. In order to repair any mechanical parts (rotor or shafts) it is necessary to remove the turbine insert. To do this, first remove all pressure from the line. Then remove the machine screws which hold the top flange in place. Lift off the flange with attached insert.

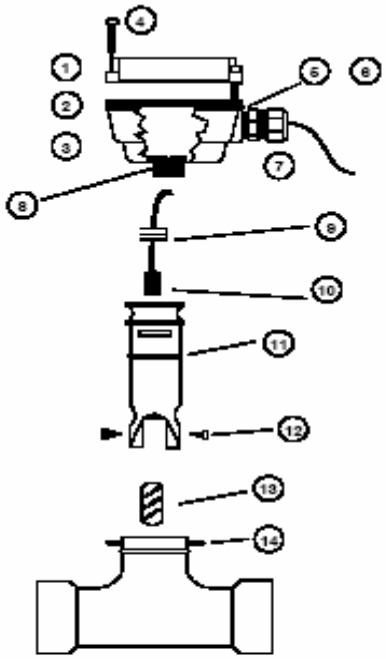
Rotor and Shaft Replacement. Examine the rotor to determine if bearings or shaft are damaged or excessively worn. The rotor should spin smoothly and freely with no visible wobble. Back and forth play should be very minor, less than 1/64". If it is necessary to replace the rotor or shafts, first back out both shafts with a screwdriver. The rotor will come free as soon as the shaft ends come free of the rotor bearings. Reverse the procedure to reinstall, taking care to maintain a small amount of free play between the shaft ends and the bearings.

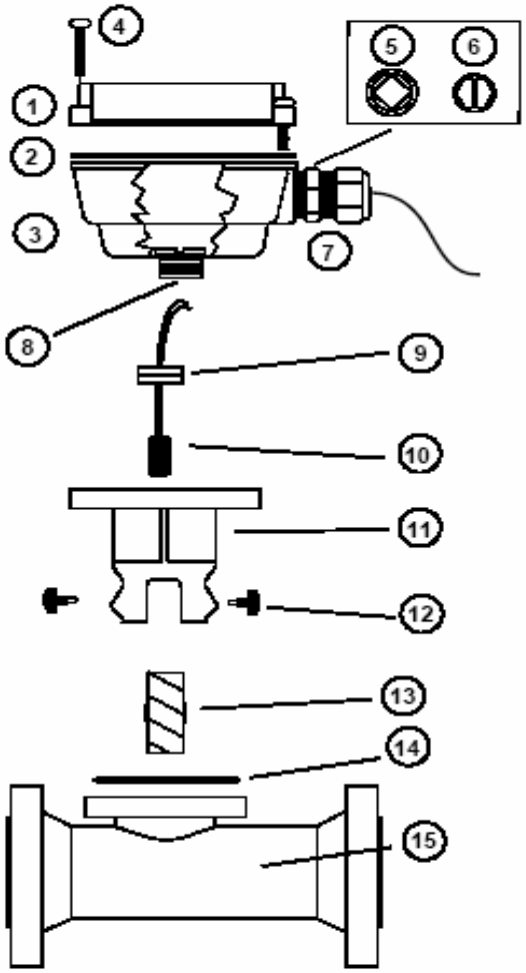
Sensor Replacement. This procedure is rarely necessary. However, certain electrical conditions can damage the sensor. To replace it, first remove any electronics module in the aluminum electronics housing. Disconnect the sensor leads from terminals on the back of the board. Unscrew the sensor retaining screw carefully and then remove the sensor by tugging gently on the sensor leads.

Electronic Module Repair. None of the electronics modules have replaceable components. Printed circuit boards must be replaced as complete units. In order to replace an electronic module, loosen the four screws which fasten each unit. Once the screws are loose, the unit will lift free from the insert housing.

WT(S) Meter Assembly:

WT(S) Parts - 2"		
1-6	Housing (see chart below)	
7	Strain relief	7655
8	Housing retaining screw	26508
9	Sensor retaining screw	25321
10	Sensor	26310
11	O-ring	25081
12	Shaft screw	16710
13	Turbine rotor	25947
14	U-clip	15527





WTS Parts - 3"-8"		
1	Upper blind housing	26181
1	PD10 divider module	26520
1	FT420 rate/total display module	26949
1	A055 4-20 mA transmitter module	26521
1	FT415 rate/total display module	26519
2	Gasket	26211
3	Lower housing	29930
4	Housing screw	26229
5	Plug, steel	26073
6	Plug, plastic	26079
7	Strain relief	7655
8	Housing retaining screw	26508
9	Sensor retainer	25321
10	Sensor	26310
11	Insert 3"-8"	16820
12	Turbine shaft screw (2)	16710
13	Turbine rotor	15316
14	0-ring	25105
15	Meter body	

SeaMetrics
20419 80th Ave. So., Kent, WA 98032 USA
Phone: 253-872-0284 Fax: 253-872-0285
www.seametrics.com 1-800-975-8153

APÉNDICE C. DATOS TÉCNICOS RELEVANTES DEL PIC16F877A

MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (VREF) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

PIC16F876A/877A REGISTER FILE MAP

Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch	CMCON	9Ch		11Ch		19Ch
CCP2CON	1Dh	CVRCON	9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
			EFh		16Fh		1EFh
accesses 70h-7Fh	7Fh	accesses 70h-7Fh	F0h FFh	accesses 70h-7Fh	170h 17Fh	accesses 70h - 7Fh	1F0h 1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F876A.
2: These registers are reserved; maintain these registers clear.

PIC16F87XA INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 1fff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 1fff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add Literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND Literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call Subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to Address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR Literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move Literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from Interrupt	2	00 0000 0000 1001		
RETLW	k	Return with Literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into Standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from Literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR Literal with W	1	11 1010 kkkk kkkk	Z	

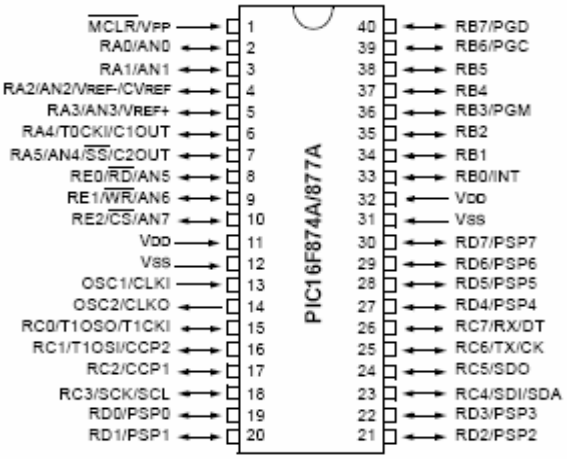
Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 module.

3: If Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

PIC16F877A Pin Diagram

40-Pin PDIP

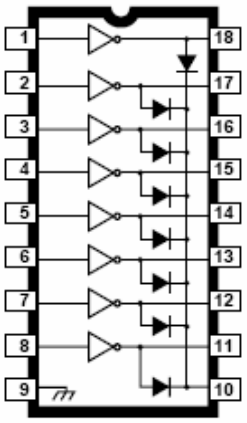


APÉNDICE D. HOJA TÉCNICA DEL ULN2803A.

2803 THRU 2824

Data Sheet 29304.3E*

HIGH-VOLTAGE, HIGH-CURRENT DARLINGTON ARRAYS



Dwg. No. A-10,322A

Note that the ULx28xxA series (dual in-line package) and ULx28xxLW series (small-outline IC package) are electrically identical and share a common terminal number assignment.

ABSOLUTE MAXIMUM RATINGS

Output Voltage, V_{CE}
- (x2803x and x2804x) ..... 50 V
- (x2823x and x2824x) ..... 95 V

Input Voltage, V_{IN} ..... 30 V

Continuous Output Current, I_C .... 500 mA

Continuous Input Current, I_{IN} ..... 25 mA

Power Dissipation, P_D
- (one Darlington pair) ..... 1.0 W
- (total package) ..... See Graph

Operating Temperature Range, T_A
- Prefix 'ULN' ..... -20°C to + 85°C
- Prefix 'ULQ' ..... -40°C to +85°C

Storage Temperature Range, T_S ..... -55°C to +150°C

Featuring continuous load current ratings to 500 mA for each of the drivers, the Series ULN28xxA/LW and ULQ28xxA/LW high-voltage, high-current Darlington arrays are ideally suited for interfacing between low-level logic circuitry and multiple peripheral power loads. Typical power loads totaling over 260 W (350 mA x 8, 95 V) can be controlled at an appropriate duty cycle depending on ambient temperature and number of drivers turned on simultaneously. Typical loads include relays, solenoids, stepping motors, magnetic print hammers, multiplexed LED and incandescent displays, and heaters. All devices feature open-collector outputs with integral clamp diodes.

The ULx2803A, ULx2803LW, ULx2823A, and ULN2823LW have series input resistors selected for operation directly with 5 V TTL or CMOS. These devices will handle numerous interface needs — particularly those beyond the capabilities of standard logic buffers.

The ULx2804A, ULx2804LW, ULx2824A, and ULN2824LW have series input resistors for operation directly from 6 V to 15 V CMOS or PMOS logic outputs.

The ULx2803A/LW and ULx2804A/LW are the standard Darlington arrays. The outputs are capable of sinking 500 mA and will withstand at least 50 V in the off state. Outputs may be paralleled for higher load current capability. The ULx2823A/LW and ULx2824A/LW will withstand 95 V in the off state.

These Darlington arrays are furnished in 18-pin dual in-line plastic packages (suffix 'A') or 18-lead small-outline plastic packages (suffix 'LW'). All devices are pinned with outputs opposite inputs to facilitate ease of circuit board layout. Prefix 'ULN' devices are rated for operation over the temperature range of -20°C to +85°C; prefix 'ULQ' devices are rated for operation to -40°C.

FEATURES

- TTL, DTL, PMOS, or CMOS Compatible Inputs
- Output Current to 500 mA
- Output Voltage to 95 V
- Transient-Protected Outputs
- Dual In-Line Package or Wide-Body Small-Outline Package

The ULx2804, ULx2823, & ULx2824 are last-time buy. Orders accepted until October 19, 2001.

x = Character to identify specific device. Characteristic shown applies to family of devices with remaining digits as shown. See matrix on next page.

Allegro MicroSystems, Inc.

DEVICE PART NUMBER DESIGNATION

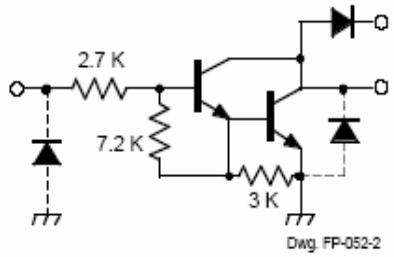
$V_{CE(MAX)}$	50 V	95 V
$I_{C(MAX)}$	500 mA	500 mA
Logic	**Part Number**	
5V TTL, CMOS	ULN2803A* ULN2803LW*	ULN2823A* ULN2823LW
6-15 V CMOS, PMOS	ULN2804A* ULN2804LW*	ULN2824A* ULN2824LW

*Also available for operation between -40°C and +85°C. To order, change prefix from 'ULN' to 'ULQ'.

The ULx2804, ULx2823, & ULx2824 are last-time buy. Orders accepted until October 19, 2001.

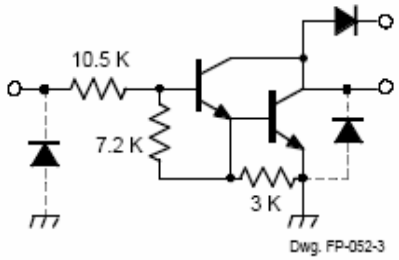
PARTIAL SCHEMATICS

ULx28x3A/LW (Each Driver)

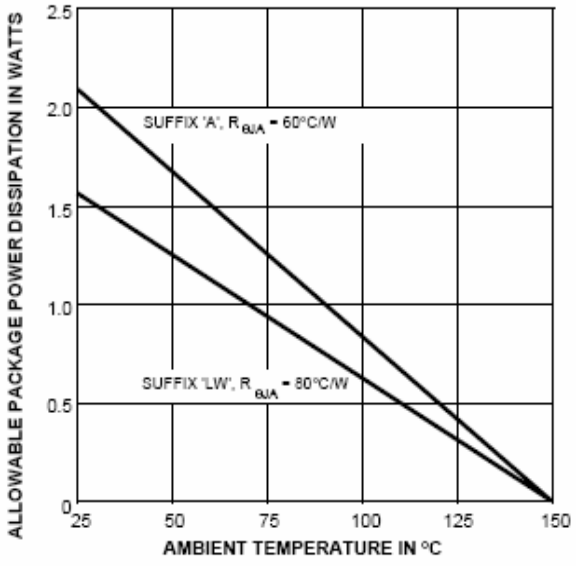


Dwg. FP-052-2

ULx28x4A/LW (Each Driver)



Dwg. FP-052-3



Dwg. GP-018B

x = Character to identify specific device. Specification shown applies to family of devices with remaining digits as shown. See matrix above.

115 Northeast Cutoff, Box 15036
Worcester, Massachusetts 01615-0036 (508) 853-5000

Types ULx2803A, ULx2803LW, ULx2804A, and ULx2804LW
ELECTRICAL CHARACTERISTICS at +25°C (unless otherwise noted).

Characteristic	Symbol	Test Fig.	Applicable Devices	Test Conditions	Limits Min.	Typ.	Max.	Units
Output Leakage Current	I_{CEX}	1A	All	V_{CE} = 50 V, T_A = 25°C	—	< 1	50	µA
				V_{CE} = 50 V, T_A = 70°C	—	< 1	100	µA
		1B	ULx2804x	V_{CE} = 50 V, T_A = 70°C, V_{IN} = 1.0 V	—	< 5	500	µA
Collector-Emitter Saturation Voltage	$V_{CE(SAT)}$	2	All	I_C = 100 mA, I_B = 250 µA	—	0.9	1.1	V
				I_C = 200 mA, I_B = 350 µA	—	1.1	1.3	V
				I_C = 350 mA, I_B = 500 µA	—	1.3	1.6	V
Input Current	$I_{IN(ON)}$	3	ULx2803x	V_{IN} = 3.85 V	—	0.93	1.35	mA
			ULx2804x	V_{IN} = 5.0 V	—	0.35	0.5	mA
				V_{IN} = 12 V	—	1.0	1.45	mA
	$I_{IN(OFF)}$	4	All	I_C = 500 µA, T_A = 70°C	50	65	—	µA
Input Voltage	$V_{IN(ON)}$	5	ULx2803x	V_{CE} = 2.0 V, I_C = 200 mA	—	—	2.4	V
				V_{CE} = 2.0 V, I_C = 250 mA	—	—	2.7	V
				V_{CE} = 2.0 V, I_C = 300 mA	—	—	3.0	V
			ULx2804x	V_{CE} = 2.0 V, I_C = 125 mA	—	—	5.0	V
				V_{CE} = 2.0 V, I_C = 200 mA	—	—	6.0	V
				V_{CE} = 2.0 V, I_C = 275 mA	—	—	7.0	V
				V_{CE} = 2.0 V, I_C = 350 mA	—	—	8.0	V
Input Capacitance	C_{IN}	—	All		—	15	25	pF
Turn-On Delay	t_{PLH}	8	All	0.5 E_{IN} to 0.5 E_{OUT}	—	0.25	1.0	µs
Turn-Off Delay	t_{PHL}	8	All	0.5 E_{IN} to 0.5 E_{OUT}	—	0.25	1.0	µs
Clamp Diode Leakage Current	I_R	6	All	V_R = 50 V, T_A = 25°C	—	—	50	µA
				V_R = 50 V, T_A = 70°C	—	—	100	µA
Clamp Diode Forward Voltage	V_F	7	All	I_F = 350 mA	—	1.7	2.0	V

Complete part number includes prefix to operating temperature range: ULN = -20°C to +85°C, ULQ = -40°C to +85°C and a suffix to identify package style: A = DIP, LW = SOIC.

The ULx2804 is last-time buy.
Orders accepted until October 19, 2001.

Types ULx2823A, ULN2823LW, ULx2824A, and ULN2824LW
ELECTRICAL CHARACTERISTICS at +25°C (unless otherwise noted).

Characteristic	Symbol	Test Fig.	Applicable Devices	Test Conditions	Limits Min.	Typ.	Max.	Units
Output Leakage Current	I_{CEX}	1A	All	V_{CE} = 95 V, T_A = 25°C	—	< 1	50	µA
				V_{CE} = 95 V, T_A = 70°C	—	< 1	100	µA
		1B	ULx2824x	V_{CE} = 95 V, T_A = 70°C, V_{IN} = 1.0 V	—	< 5	500	µA
Collector-Emitter Saturation Voltage	$V_{CE(SAT)}$	2	All	I_C = 100 mA, I_B = 250 µA	—	0.9	1.1	V
				I_C = 200 mA, I_B = 350 µA	—	1.1	1.3	V
				I_C = 350 mA, I_B = 500 µA	—	1.3	1.6	V
Input Current	$I_{IN(ON)}$	3	ULx2823x	V_{IN} = 3.85 V	—	0.93	1.35	mA
			ULx2824x	V_{IN} = 5.0 V	—	0.35	0.5	mA
				V_{IN} = 12 V	—	1.0	1.45	mA
	$I_{IN(OFF)}$	4	All	I_C = 500 µA, T_A = 70°C	50	65	—	µA
Input Voltage	$V_{IN(ON)}$	5	ULx2823x	V_{CE} = 2.0 V, I_C = 200 mA	—	—	2.4	V
				V_{CE} = 2.0 V, I_C = 250 mA	—	—	2.7	V
				V_{CE} = 2.0 V, I_C = 300 mA	—	—	3.0	V
			ULx2824x	V_{CE} = 2.0 V, I_C = 125 mA	—	—	5.0	V
				V_{CE} = 2.0 V, I_C = 200 mA	—	—	6.0	V
				V_{CE} = 2.0 V, I_C = 275 mA	—	—	7.0	V
				V_{CE} = 2.0 V, I_C = 350 mA	—	—	8.0	V
Input Capacitance	C_{IN}	—	All		—	15	25	pF
Turn-On Delay	t_{PLH}	8	All	0.5 E_{IN} to 0.5 E_{OUT}	—	0.25	1.0	µs
Turn-Off Delay	t_{PHL}	8	All	0.5 E_{IN} to 0.5 E_{OUT}	—	0.25	1.0	µs
Clamp Diode Leakage Current	I_R	6	All	V_R = 95 V, T_A = 25°C	—	—	50	µA
				V_R = 95 V, T_A = 70°C	—	—	100	µA
Clamp Diode Forward Voltage	V_F	7	All	I_F = 350 mA	—	1.7	2.0	V

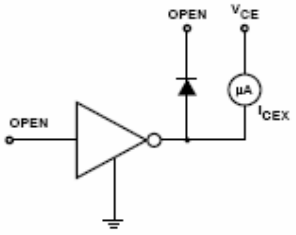
Complete part number includes prefix to operating temperature range: ULN = -20°C to +85°C, ULQ = -40°C to +85°C and a suffix to identify package style: A = DIP, LW = SOIC. Note that the ULQ2823LW and ULQ2824LW are not presently available.

The ULx2823 & ULx2824 are last-time buy.
Orders accepted until October 19, 2001.

Allegro MicroSystems, Inc.
115 Northeast Cutoff, Box 15036
Worcester, Massachusetts 01615-0036 (508) 853-5000

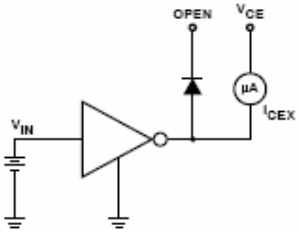
TEST FIGURES

FIGURE 1A



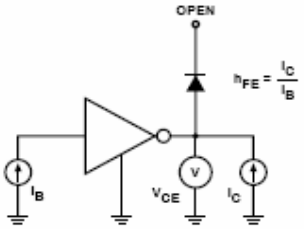
Dwg. No. A-9729A

FIGURE 1B



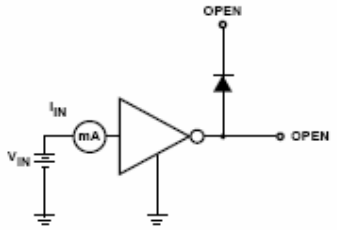
Dwg. No. A-9730A

FIGURE 2



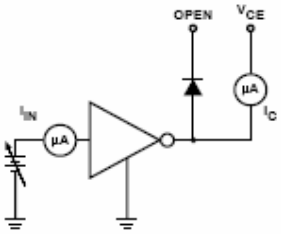
Dwg. No. A-9731A

FIGURE 3



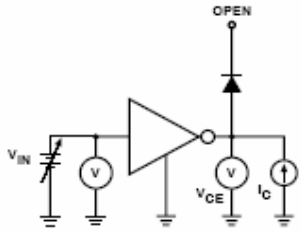
Dwg. No. A-9732A

FIGURE 4



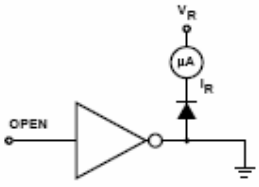
Dwg. No. A-9733A

FIGURE 5



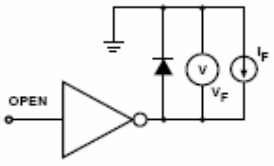
Dwg. No. A-9734A

FIGURE 6



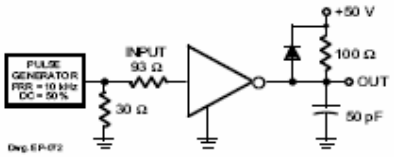
Dwg. No. A-9735A

FIGURE 7



Dwg. No. A-9736A

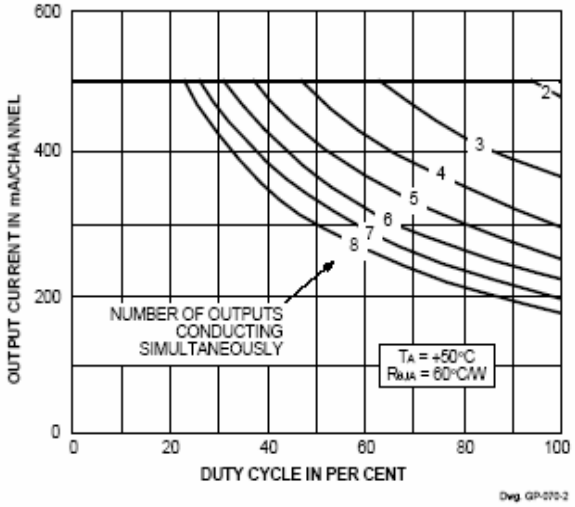
FIGURE 8



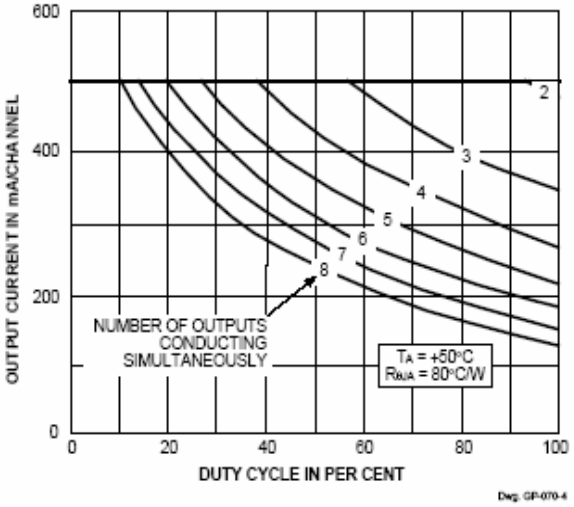
Dwg. EP-012

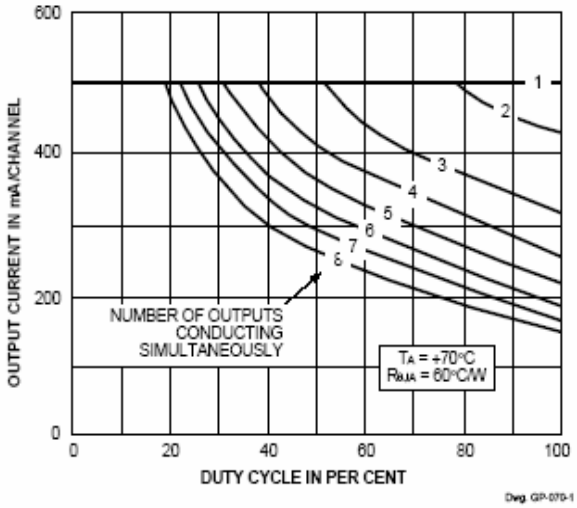
	V_{IN}
ULx28x3x	3.5 V
ULx28x4x	12 V

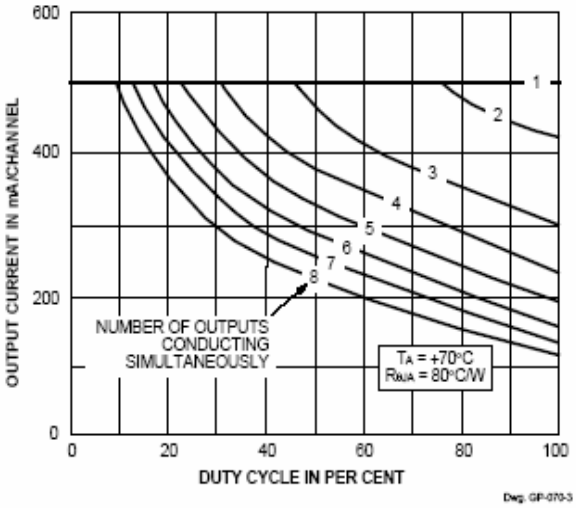
ALLOWABLE COLLECTOR CURRENT AS A FUNCTION OF DUTY CYCLE
ULx28xxA



ALLOWABLE COLLECTOR CURRENT AS A FUNCTION OF DUTY CYCLE
ULx28xxLW





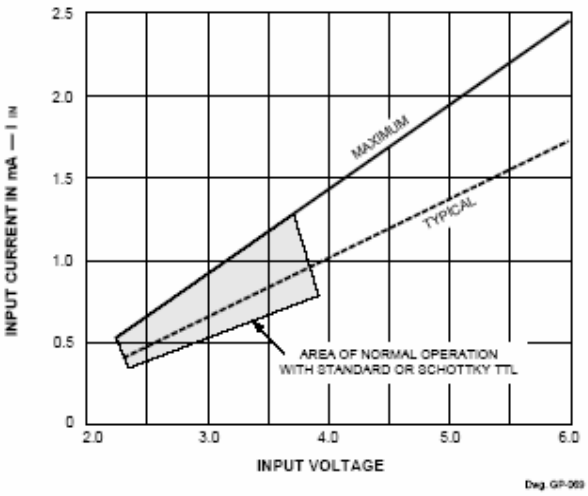


x = Characters to identify specific device. Specification shown applies to family of devices with remaining digits as shown.

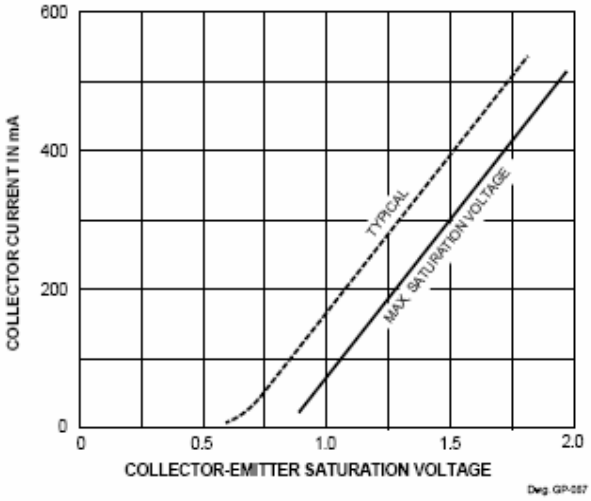
Allegro MicroSystems, Inc.
115 Northeast Cutoff, Box 15036
Worcester, Massachusetts 01615-0036 (508) 853-5000

INPUT CURRENT AS A FUNCTION OF INPUT VOLTAGE

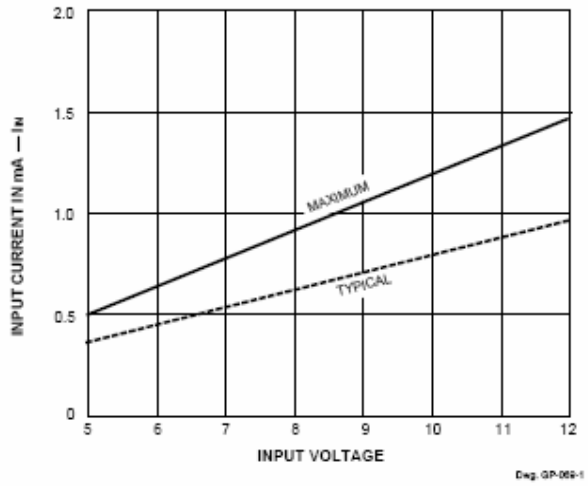
ULx28x3x



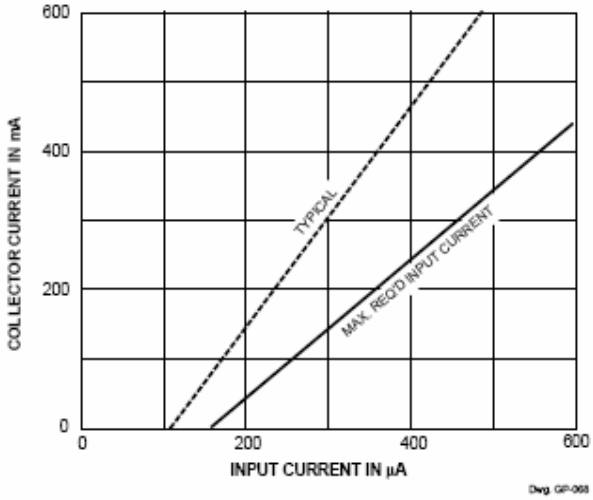
SATURATION VOLTAGE AS A FUNCTION OF COLLECTOR CURRENT



ULx28x4x



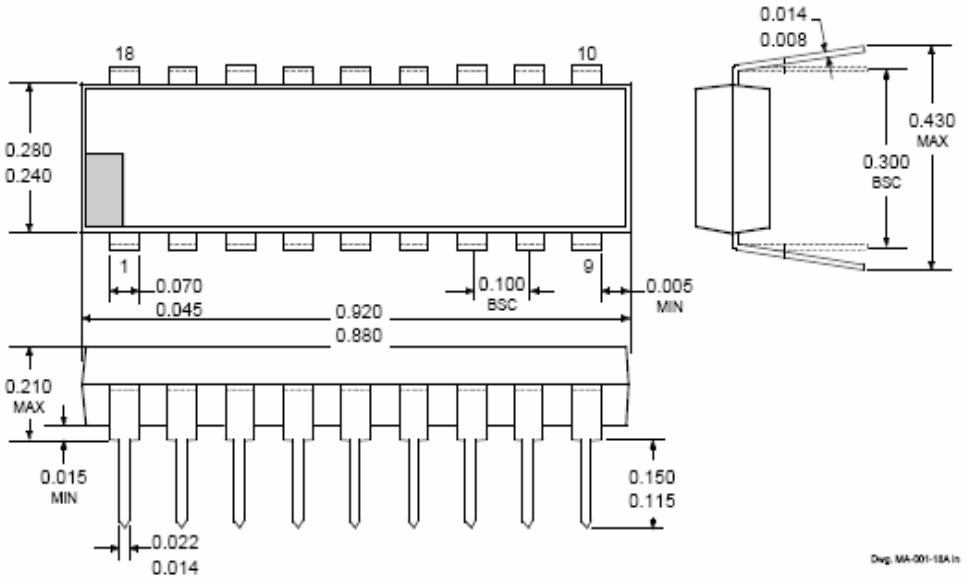
COLLECTOR CURRENT AS A FUNCTION OF INPUT CURRENT



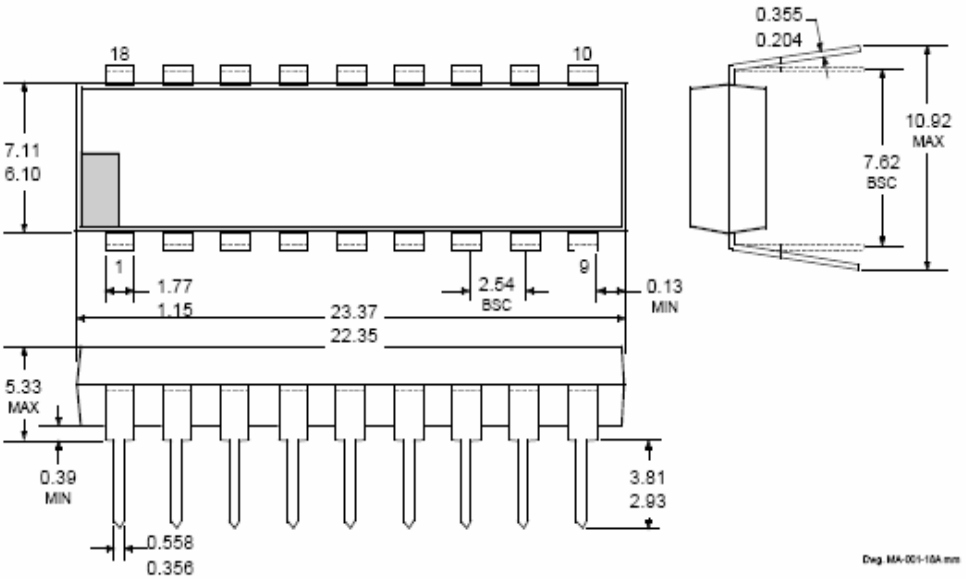
x = Characters to identify specific device. Characteristic shown applies to family of devices with remaining digits as shown.

PACKAGE DESIGNATOR "A" DIMENSIONS

Dimensions in Inches
(controlling dimensions)



Dimensions in Millimeters
(for reference only)

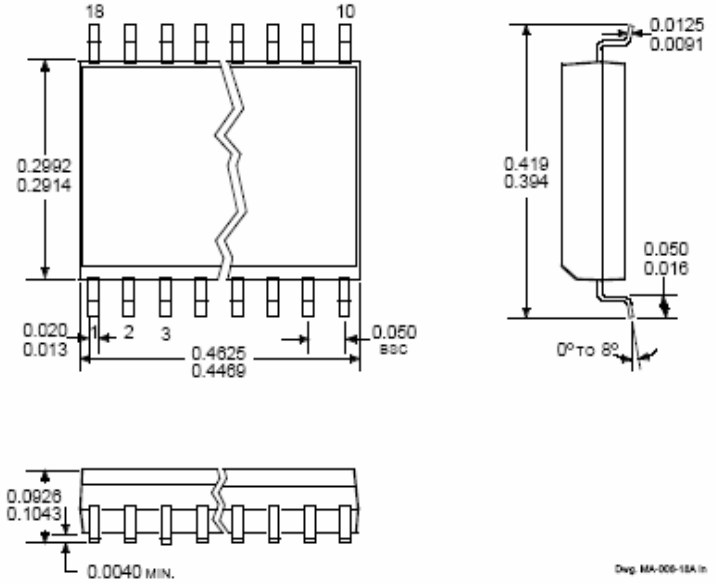


NOTES: 1. Exact body and lead configuration at vendor's option within limits shown.
2. Lead spacing tolerance is non-cumulative.
3. Lead thickness is measured at seating plane or below.

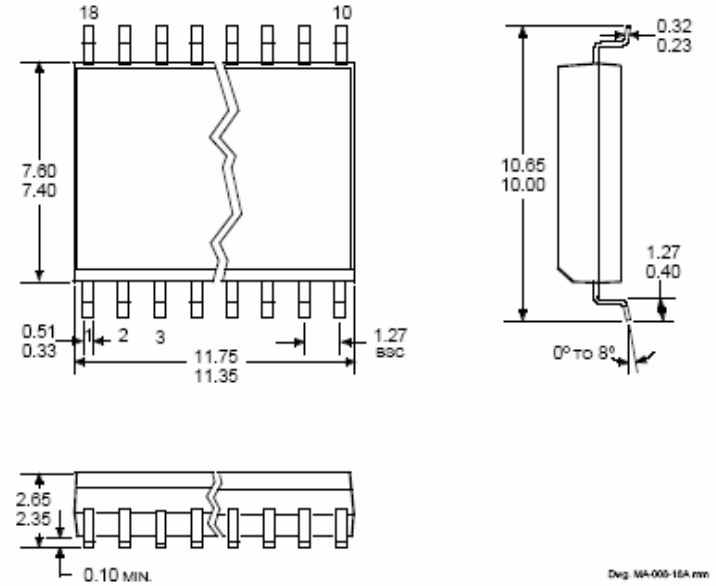
Allegro MicroSystems, Inc.
115 Northeast Cutoff, Box 15036
Worcester, Massachusetts 01615-0036 (508) 853-5000

PACKAGE DESIGNATOR "LW" DIMENSIONS

Dimensions in Inches
(for reference only)



Dimensions in Millimeters
(controlling dimensions)



NOTES: 1. Exact body and lead configuration at vendor's option within limits shown.
2. Lead spacing tolerance is non-cumulative.

APÉNDICE E. HOJA TÉCNICA TELÉFONO MÓVIL NOKIA 5140

The Nokia 5140 imaging phone combines high-speed data (EDGE) capabilities with a variety of features. Messaging features include WAP 2.0 (XHTML) browsing, instant messaging (IM), and push-to-talk capabilities. Lifestyle features include a digital compass, thermometer, FM stereo radio, and flashlight.

Nokia 5140 Technical Specs
Operating System:
Nokia OS
Developer Platform:
Series 40 Developer Platform 2.0
Java Technology:
CLDC 1.1
Wireless Messaging API (JSR-120)
Mobile Media API (JSR-135)
JTWI (JSR-185)
MIDP 2.0
Nokia UI API
Browser:
WAP 2.0
XHTML over TCP/IP
Messaging:
MMS+SMIL
SMS
UAProfile:
Profile 1
Digital Rights Management:
OMA DRM v1.0
Delivery Method:
MMS
WAP Download
Sound Formats:
AMR (NB-AMR)
MIDI tones (poly 16)
Functionality:
GSM 1800
GSM 1900
GSM 900
Regional Availability:

Tools

The main Tools & SDKs page -www.forum.nokia.com/tools provides all the tools & SDKs in an easy to access form. The page also includes Getting Started -articles giving an overview of the tools and SDKs.

Nokia Developer's Suite 3.0 for J2ME™

Nokia Developer's Suite for the Java™ Platform, Micro Edition, Version 3.0 provides developers with utilities for creating and deploying MIDP 1.0 and MIDP 2.0 applications. New in this version are Nokia Developer's Suite for J2ME integration support for Eclipse 2.1.x and 3.0 and an Eclipse plug-in for integrating Nokia SDKs into the Eclipse environment.

Nokia Mobile Internet Toolkit 4.1

NMIT is a content authoring tool that supports the creation of Mobile Internet content like XHTML, WAP, and MMS (including SMIL transitions and timeline). Content can be previewed on supported SDKs. DRM protection can be applied to such content.

Nokia Series 40 Theme Studio

Nokia Series 40 Theme Studio is a PC based tool for creating User Interface themes for compatible Series 40 handsets. Theme developers can create, view, compare and save themes. The created theme packages can be downloaded to compatible handsets using methods supported by both the handsets and the PC (bluetooth, IrDa, Cable) or over the air (OTA).

Recommended SDK(s)

Series 40 Platform SDKs

The Series 40 Platform 3rd Edition SDK is the reference implementation SDK for the Series 40 Platform 3rd Edition. The SDK enables developers to quickly and efficiently run and test Java applications as well as browser and Multimedia Messaging Service (MMS) content. The product consists of an emulator, Java APIs, documentation, sample code and debugging tools.

Africa
Americas
Asia-Pacific
Europe

Screen Display:

Color Depth: 12 bit
Resolution: 128 x 128

Physical Descriptions:

Dimensions: 106.5 x 47 x 24 mm
Weight: 101 g

Memory:

Heap size: 500 KB
Shared Memory for Storage: 977 KB
Max JAR Size: 125 KB

Keypad Descriptions:

3 labeled soft keys
4-way scrolling
Grid key mat

Video Support:

3GPP formats (H.263)

Network Data Support:

CSD
EGPRS
GPRS
HSCSD

PC Connectivity:

Infrared
USB

Extra Features:

Flashlight
FM radio
Handsfree speaker
Instant Messaging
Thermometer
VGA camera

Consumer link:

Product Home Page

Developer link:

Developer Home Page

Documents

Developer Platform 2.0: Known Issues v1.18
This updated document describes the known issues for Developer Platform 2.0. One new Browsing known issue and one new Symbian known issue have been added.

Series 40 Developer Platform: Introductory White Paper
This document introduces Series 40 Developer Platform; it describes the user functionality provided by the platform and provides an overview of the options for developing applications and content.

Series 40 Developer Platform: FAQ
This document answers the most frequently asked questions from developers and business managers regarding the Series 40 Developer Platform, including recent enhancements.

MIDP 2.0: Introduction v1.1
This updated document provides an introduction to MIDP 2.0. The document addresses the majority of the new features, covering the most important changes from MIDP 1.0 as well as introducing the new classes and packages.

The availability of the product and its features depends on regional availability and service providers. Please contact your service provider and your Nokia dealer for further information. These specifications are subject to change without notice.

APÉNDICE F. DATOS TÉCNICOS TRANSCEIVER TDFS6XX.

VISHAY

TFDU6100/TFDS6500/TFDT6500

Vishay Telefunken

5 V Fast Infrared Transceiver Module Family (FIR, 4 Mbit/s)

MEMBER IrDA

Description

The TFDU6100, TFDS6500, and TFDT6500 are a family of low–power infrared transceiver modules compliant to the IrDA 1.2 standard for fast infrared (FIR) data communication, supporting IrDA speeds up to 4.0 Mbit/s, HP-SIR, Sharp ASK and carrier based remote control modes up to 2 MHz. Integrated within the transceiver modules are a photo PIN diode, infrared emitter (IRED), and a low–power CMOS control IC to provide a total front–end solution in a single package. Vishay Telefunken's FIR transceivers are available in three package options, including our Baby Face package (TFDU6100), the smallest FIR transceiver available on the market. This wide selection provides flexibility for a variety of applications and space constraints. The transceivers are capable of directly interfacing with a wide variety of I/O chips which perform the modulation/ demodulation function, including National Semiconductor's PC87338, PC87108 and PC87109, SMC's FDC37C669, FDC37N769 and CAM35C44, and Hitachi's SH3. At a minimum, a current–limiting resistor in series with the infrared emitter and a V_{CC} bypass capacitor are the only external components required to implement a complete solution.

Features

- Compliant to IrDA 1.2 (Up to 4 Mbit/s), HP–SIR®, Sharp ASK® and TV Remote
- For 4.5 V to 5.5 V Operating Voltage
- Low–Power Consumption (5 mA Supply Current)
- Power Shutdown Mode (35 µA Shutdown Current)
- Three Surface Mount Package Options
  - Universal (9.7 × 4.7 × 4.0 mm)
  - Side View (13.0 × 5.95 × 5.3 mm)
  - Top View (13.0 × 7.6 × 5.95 mm)
- Baby Face (Universal) Package Capable of Surface Mount Solderability to Side and Top View Orientation
- Directly Interfaces with Various Super I/O and Controller Devices
- Built–In EMI Protection – No External Shielding Necessary
- Few External Components Required
- Backward Compatible to all Telefunken SIR and FIR Infrared Transceivers

Applications

- Notebook Computers, Desktop PCs, Palmtop Computers (Win CE, Palm PC), PDAs
- Digital Still and Video Cameras
- Printers, Fax Machines, Photocopiers, Screen Projectors
- Telecommunication Products (Cellular Phones, Pagers)
- Internet TV Boxes, Video Conferencing Systems
- External Infrared Adapters (Dongles)
- Medical and Industrial Data Collection Devices

Package Options

TFDU6100
Baby Face (Universal)

TFDS6500
Side View

TFDT6500
Top View

Ordering Information

Part Number	Qty / Reel	Description
TFDU6100–TR3	1000 pcs	Oriented in carrier tape for side view surface mounting
TFDU6100–TT3	1000 pcs	Oriented in carrier tape for top view surface mounting
TFDS6500–TR3	750 pcs	
TFDT6500–TR3	750 pcs	

Functional Block Diagram

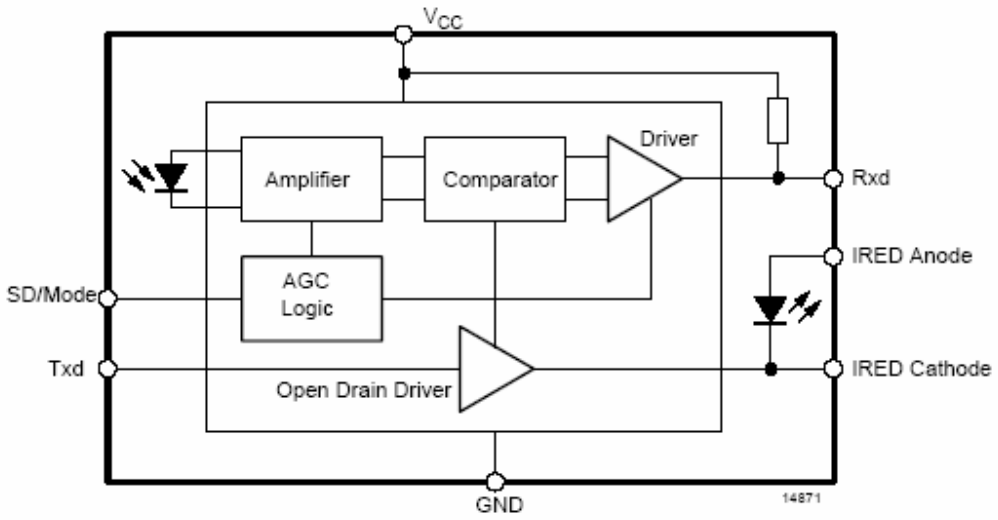


Figure 1. Functional Block Diagram

Pin Description

Pin Number		Function	Description	I/O	Active
"U" and "T" Option	"S" Option				
1	8	IRED Anode	IRED anode, to be externally connected to V_{CC} through a current control resistor		
2	1	IRED Cathode	IRED cathode, internally connected to driver transistor		
3	7	Txd	Transmit Data Input	I	HIGH
4	2	Rxd	Received Data Output, push–pull CMOS driver output capable of driving a standard CMOS or TTL load. No external pull–up or pull–down resistor is required (pin is floating when device is in shutdown mode)	O	LOW
5	6	SD/Mode	Shutdown/Mode	I	HIGH
6	3	V_{CC}	Supply Voltage		
7	5	NC	Do not connect		
8	4	GND	Ground		

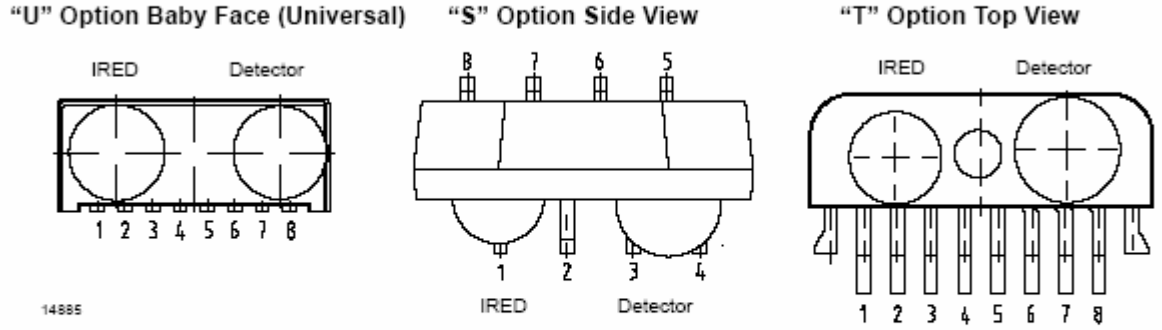


Figure 2. Pinnings

Absolute Maximum Ratings

Reference point Pin GND unless otherwise noted.
Typical values are for DESIGN AID ONLY, not guaranteed nor subject to production testing.

Parameters	Test Conditions	Symbol	Min.	Typ.	Max.	Unit
Supply Voltage Range		V_{CC}	– 0.5		6	V
Power Dissipation	See Derating Curve	P_D			350	mW
Junction Temperature		T_J			125	°C
Ambient Temperature Range (Operating)		T_{amb}	–25		+85	°C
Storage Temperature Range		T_{stg}	–25		+85	°C
Soldering Temperature	See Recommended Solder Profile (see figure 9)				240	°C
Average Output Current		I_{IRED} (DC)			130	mA
Repetitive Pulsed Output Current	<90 µs, t_{on} <20%	I_{IRED} (RP)			600	mA
IRED Anode Voltage		V_{IREDA}	– 0.5		V_{CC}+0.5	V
Transmitter Data Input Voltage		V_{Txd}	– 0.5		V_{CC}+0.5	V
Receiver Data Output Voltage		V_{Rxd}	– 0.5		V_{CC}+0.5	V
Virtual Source Size	Method: (1–1/e) encircled energy	d	2.5	2.8		mm
Maximum Intensity for Class 1 Operation of IEC825–1 or EN60825–1 (worst case IrDA FIR pulse pattern)	EN60825, 1997				320	mW/sr

Electrical Characteristics

T_{amb} = 25°C, V_{CC} = 5.0 V unless otherwise noted.
Typical values are for DESIGN AID ONLY, not guaranteed nor subject to production testing.

Parameters	Test Conditions / Pins	Symbol	Min.	Typ.	Max.	Unit
Transceiver						
Supply Voltage		V_{CC}	4.5	5	5.5	V
Dynamic Supply Current	SD = Low Receive mode only. In transmit mode, add additional 100 mA (typ) for IRED current.	I_{CC}		5	7	mA
Standby Supply Current	SD = V_{CC} –0.5	I_{SD}		35	100	µA
Operating Temperature Range		T_A	–25		+85	°C
Output Voltage Low	I_{OL} = 2.5 mA	V_{OL}		0.3	0.5	V
Output Voltage High	I_{OH} = –2.5 mA	V_{OH}	V_{CC}–0.5			V
Input Voltage Low (T_{xd})		V_{IL}	0		0.8	V
Input Voltage High (T_{xd})		V_{IH}	3.5			V
Input Voltage Low (SD/Mode)		V_{IL}	0		0.8	V
Input Voltage High (SD/Mode)		V_{IH}	V_{CC} –0.5			V
Input Leakage Current		I_L	–10		+10	µA
Input Capacitance		C_I			5	pF

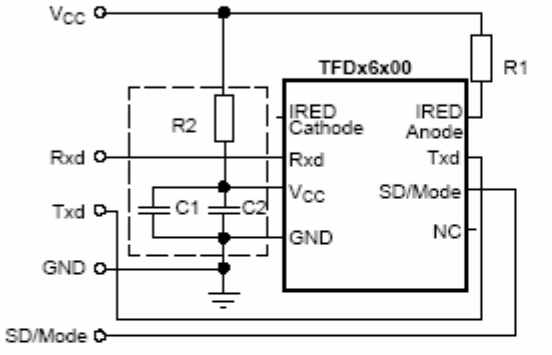
Optoelectronic Characteristics

T_{amb} = 25°C, V_{CC} = 5.0 V unless otherwise noted.
Typical values are for DESIGN AID ONLY, not guaranteed nor subject to production testing.

Parameters	Test Conditions	Symbol	Min.	Typ.	Max.	Unit
Receiver						
Minimum Detection Threshold Irradiance	9.6 kbit/s to 115.2 kbit/s, λ = 850 nm – 900 nm	E_e		25	35	mW/m²
	1.152 Mbit/s to 4 Mbit/s, λ = 850 nm – 900 nm	E_e		70	80	mW/m²
Maximum Detection Threshold Irradiance		E_e	5	10		kW/m²
Logic LOW Receiver Input Irradiance		E_e	4			mW/m²
Rise Time of Output Signal	10% to 90%, @2.2 kΩ, 15pF	$t_{r\ (Rxd)}$	10		40	ns
Fall Time of Output Signal	90% to 10%, @2.2 kΩ, 15pF	$t_{f\ (Rxd)}$	10		40	ns
Rxd Pulse Width of Output Signal, 50%	Input pulse length 20 µs, 9.6 kbit/s	t_{PW}	0.8		20	µs
	Input pulse length 125 ns, 4.0 Mbit/s mode	t_{PW}	60		165	ns
	Input pulse length 250 ns, 4.0 Mbit/s mode (double pulse)	t_{PW}	185		290	ns
Jitter, Leading Edge	Input Irradiance = 90 mW/m², 4.0 Mbit/s mode				10	ns
Latency		t_L			120	µs
Transmitter						
IRED Operating Current	R1 = 7.2 Ω, V_{CC} = 5.0 V	I_D		0.4	0.55	A
Output Radiant Intensity	Vcc = 5.0 V, α = 0°, 15° Txd = Low or SD = High (Receiver is inactive as long as SD = High) R1 = 7.2 Ω	I_e			0.04	mW/sr
	Vcc = 5.0 V, α = 0°, 15° Txd = High, SD = Low, R1 = 7.2 Ω	I_e	100	140	320	mW/sr
Radiant Intensity, Half – Intensity Angle		$\alpha_{1/2}$		±24		°
Peak – Emission Wavelength		λ_p	880		900	nm
Optical Rise Time, Fall Time		t_{ropt}, t_{fopt}	10		40	ns
Optical Overshoot					25	%

Recommended Circuit Diagram

The only required component for designing an IrDA 1.2 compatible design using Telefunken FIR transceivers is a current limiting resistor, R1, to the IRED. However, depending on the entire system design and board layout, additional components may be required (see figure 3).



Note: Outlined components are optional depending on the quality of the power supply.

14872

Figure 3. Recommended Application Circuit

Telefunken FIR transceivers integrate a sensitive receiver and a built–in power driver. The combination of both needs a careful circuit board layout. The use of thin, long resistive and inductive wiring should be avoided. The inputs (Txd, SD/Mode) and the output Rxd should be directly (DC) coupled to the I/O circuit.

R1 is used for controlling the current through the IR emitter. For increasing the output power of the IRED, the value of the resistor should be reduced. Similarly, to reduce the output power of the IRED, the value of the resistor should be increased. For typical values of R1 see figure 4. For IrDA compliant operation, a current control resistor of 7.2 Ω is recommended. The upper drive current limitation is dependent on the duty cycle and is given by the absolute maximum ratings on the data sheet.

R2, C1 and C2 are optional and dependent on the quality of the supply voltage V_{CC} and injected noise. An unstable power supply with dropping voltage during transmission may reduce sensitivity (and transmission range) of the transceiver.

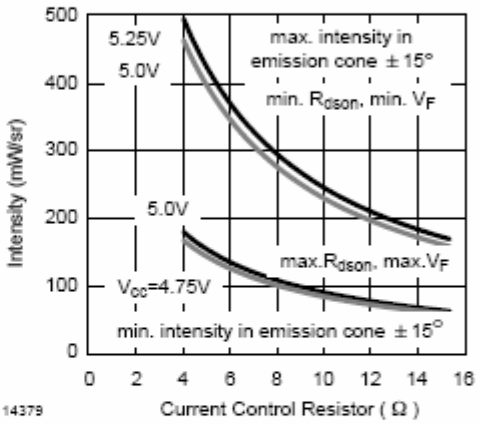


Figure 4. Intesity I_e vs. Current Control Resistor R1

The placement of these parts is critical. It is strongly recommended to position C2 as near as possible to the transceiver power supply pins. A tantalum capacitor should be used for C1 while a ceramic capacitor is used for C2. Also, when connecting the described circuit to the power supply, low impedance wiring should be used.

Table 1. Recommended Application Circuit Components

Component	Recommended Value
C1	4.7 µF, Tantalum
C2	0.1 µF, Ceramic
R1	7.2 Ω , 0.25 W (recommend using two 3.6 Ω, 0.125 W resistors in series)
R2	47 Ω , 0.125 W

Mode Switching

The TFDU6100, TFDS6500 and TFDT6500 powers on with a default of low frequency mode.

The low frequency mode covers speeds up to 115.2 kbit/s. Signals with higher data rates should be detected in the high frequency mode. Lower frequency data can also be received in the high frequency mode but with reduced sensitivity. To switch the transceivers from low frequency mode to the 4.0 Mbit/s mode and vice versa, the programming sequences described below are required.

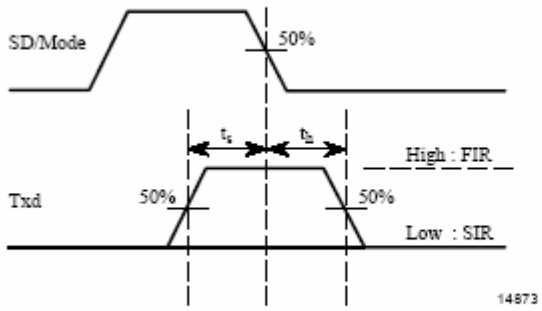


Figure 5. Mode Switching Timing Diagram

Setting to the High Bandwidth Mode (0.576 Mbit/s to 4.0 Mbit/s)

1. Set SD/MODE input to logic "HIGH".
2. Set Txd input to logic "HIGH". Wait $t_s \geq 200$ ns.
3. Set SD/MODE to logic "LOW" (this negative edge latches state of Txd, which determines speed setting).
4. After waiting $t_h \geq 200$ ns Txd can be set to logic "LOW". The hold time of Txd is limited by the maximum allowed pulse length.

Txd is now enabled as normal Txd input for the high bandwidth mode.

Setting to the Lower Bandwidth Mode (2.4 kbit/s to 115.2 kbit/s)

1. Set SD/MODE input to logic "HIGH".
2. Set Txd input to logic "LOW". Wait $t_s \geq 200$ ns.
3. Set SD/MODE to logic "LOW" (this negative edge latches state of Txd, which determines speed setting).
4. Txd must be held for $t_h \geq 200$ ns.

Txd is now enabled as normal Txd input for the lower bandwidth mode.

APÉNDICE G. DATOS TÉCNICOS CODIFICADOR MCP2120.

MICROCHIP

MCP2120

Infrared Encoder/Decoder

FEATURES

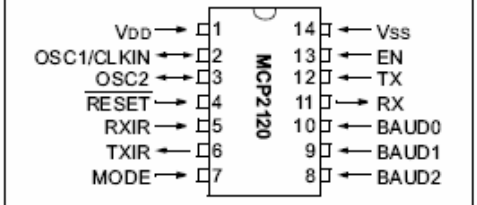
- Supports with IrDA® Physical Layer Specification (version 1.3)
- UART to IR Encoder/Decoder
  - Interfaces with IrDA Compliant Transceivers
  - Used with any UART, including standard 16550 UART and microcontroller UART
- Transmit/Receive formats supported:
  - 1.63 µs
- Hardware or Software Baud rate selection
  - Up to IrDA standard 115.2 kbaud operation
  - Up to 312.5 kbaud operation (at 20 MHz)
  - Low power mode

CMOS TECHNOLOGY

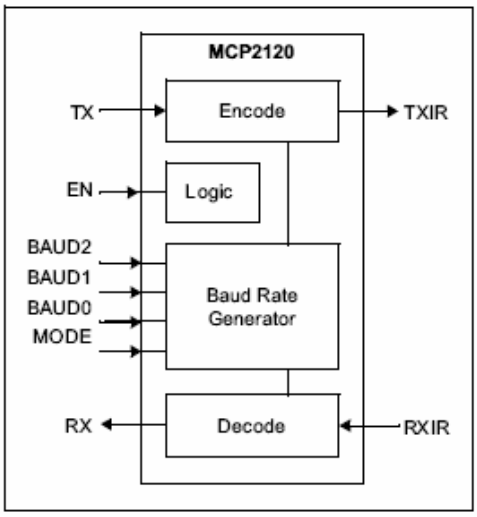
- Low-power, high-speed CMOS technology
- Fully static design
- Low voltage operation
- Commercial and Industrial temperature ranges
- Low power consumption
  - < 1 mA @ 3.3V, 8 MHz (typical)
  - 3 µA typical @ 5.0V when disabled

PIN DIAGRAMS

PDIP, SOIC



BLOCK DIAGRAM



IrDA is a registered trademark of the Infrared Data Association.

1.0 DEVICE OVERVIEW

This document contains device specific information for the following device:

- MCP2120

This device is a low-cost, high-performance, fully-static infrared encoder/decoder. This device sits between a UART and an infrared (IR) optical transceiver.

The data received from a standard UART is encoded (modulated), and output as electrical pulses to the IR Transceiver. The IR Transceiver also receives data which it outputs as electrical pulses. The MCP2120 decodes (demodulates) these electrical pulses and then the data is transmitted by the MCP2120 UART. This modulation and demodulation method is performed in accordance with the IrDA standard.

Typically a microcontroller interfaces to the IR encoder/decoder.

Infrared communication is a wireless two-way data connection using infrared light generated by low-cost transceiver signaling technology. This provides reliable communication between two devices.

Infrared technology offers:

- Universal standard for connecting portable computing devices
- Easy, effortless implementation
- Economical alternative to other connectivity solutions
- Reliable, high speed connection
- Safe to use in any environment; can even be used during air travel
- Eliminates the hassle of cables
- Allows PC's and non-PC's to communicate to each other
- Enhances mobility by allowing users to easily connect

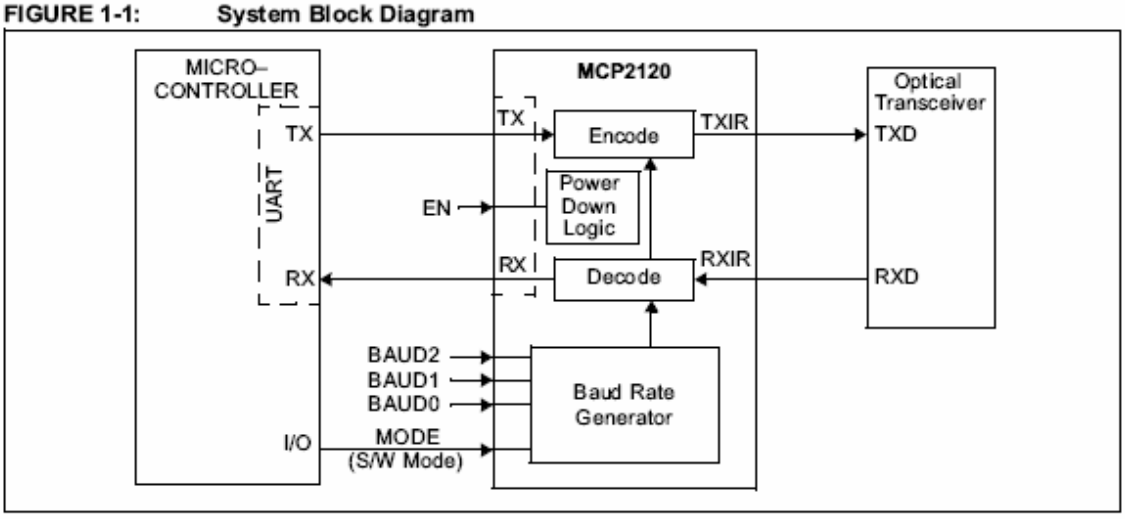
1.1 Applications

The MCP2120 is a stand–alone IrDA Encoder/Decoder product. Figure 1-1 shows a typical application block diagram. Table 1-2 shows the pin definitions in the user (normal) mode of operation.

TABLE 1-1: MCP2120 FEATURES OVERVIEW

Features	MCP2120
Serial Communications:	UART, IR
Baud Rate Selection:	Hardware/Software
Low Power Mode:	Yes
Resets: (and Delays)	Wake-up (DRT)
Packages:	14-pin DIP 14-pin SOIC

FIGURE 1-1: System Block Diagram



2.4.1 BAUD RATE

The baud rate for the MCP2120 can be configured either by the state of three hardware pins (BAUD2, BAUD1, and BAUD0) or through software selection.

2.4.1.1 Hardware Selection

Three device pins are used to select the baud rate that the MCP2120 will transmit and receive data. These pins are called BAUD2, BAUD1, and BAUD0. There is one pin state (device mode) where the application software can specify the baud rate. Table 2-1 shows the baud rate configurations.

TABLE 2-1: HARDWARE BAUD RATE SELECTION VS. FREQUENCY

BAUD2:BAUD0	Frequency (MHz)							Bit Rate
	0.6144 (1)	2.000	3.6864	4.9152	7.3728	14.7456 (2)	20.000 (2)	
000	800	2604	4800	6400	9600	19200	26042	Fosc / 768
001	1600	5208	9600	12800	19200	38400	52083	Fosc / 384
010	3200	10417	19200	25600	38400	78600	104167	Fosc / 192
011	4800	15625	28800	38400	57600	115200	156250	Fosc / 128
100	9600	31250	57600	78600	115200	230400	312500	Fosc / 64

Note 1: An external clock is recommended for frequencies below 2 MHz.

2: For frequencies above 7.5 MHz, the TXIR pulse width (parameter IR121) will be shorter than the minimum pulse width of 1.6 µs in the IrDA standard specification.

2.4.1.2 Software Selection

When the BAUD2:BAUD0 pins are configured as '111' the MCP2120 defaults to a baud rate of Fosc / 768.

To place the MCP2120 into Command Mode, the MODE pin must be at a low level. When in this mode, any data that is received by the MCP2120's UART is "echoed" back to the controller and no encoding/decoding occurs. The echoed data will be skewed less than 1 bit time (see parameter IR141). When the MODE pin goes high, the device is returned to Data Mode where the encoder/decoder is in operation.

Table 2-2 shows the software hex commands to configure the MCP2120's baud rate.

TABLE 2-2: SOFTWARE BAUD RATE SELECTION VS. FREQUENCY

Hex Command (3, 4)	Frequency (MHz)							Bit Rate
	0.6144 (1)	2.000	3.6864	4.9152	7.3728	14.7456 (2)	20.000 (2)	
0x87	800	2604	4800	6400	9600	19200	26042	Fosc / 768
0x8B	1600	5208	9600	12800	19200	38400	52083	Fosc / 384
0x85	3200	10417	19200	25600	38400	78600	104167	Fosc / 192
0x83	4800	15625	28800	38400	57600	115200	156250	Fosc / 128
0x81	9600	31250	57600	78600	115200	230400	312500	Fosc / 64

Note 1: An external clock is recommended for frequencies below 2 MHz.

2: For frequencies above 7.3728 MHz, the TXIR pulse width (parameter IR121) will be shorter than the 1.6 µs IrDA standard specification.

3: Command 0x11 is used to change to the new baud rate.

4: All other command codes are reserved for possible future use.

2.5 Modulation

When the UART receives data to be transmitted, the data needs to be modulated. This modulated signal drives the IR transceiver module. Figure 2-2 shows the encoding of the modulated signal.

Each bit time is comprised of 16-bit clocks. If the value to be transmitted (as determined by the TX pin) is a logic low, then the TXIR pin will output a low level for 7-bit clock cycles, a logic high level for 3-bit clock cycles, and then the remaining 6-bit clock cycles will be low. If the value to transmit is a logic high, then the TXIR pin will output a low level for the entire 16-bit clock cycles.

2.6 Demodulation

The modulated signal from the IR transceiver module needs to be demodulated to form the received data. As demodulation occurs, the bit value is placed on the RX pin in UART format. Figure 2-3 shows the decoding of the modulated signal.

Each bit time is comprised of 16 bit clocks. If the value to be received is a logic low, then the RXIR pin will be a low level for the first 3-bit clock cycles, and then the remaining 13-bit clock cycles will be high. If the value to be received is a logic high, then the RXIR pin will be a high level for the entire 16-bit clock cycles. The level on the RX pin will be in the appropriate state for the entire 16 clock cycles.

FIGURE 2-2: Encoding

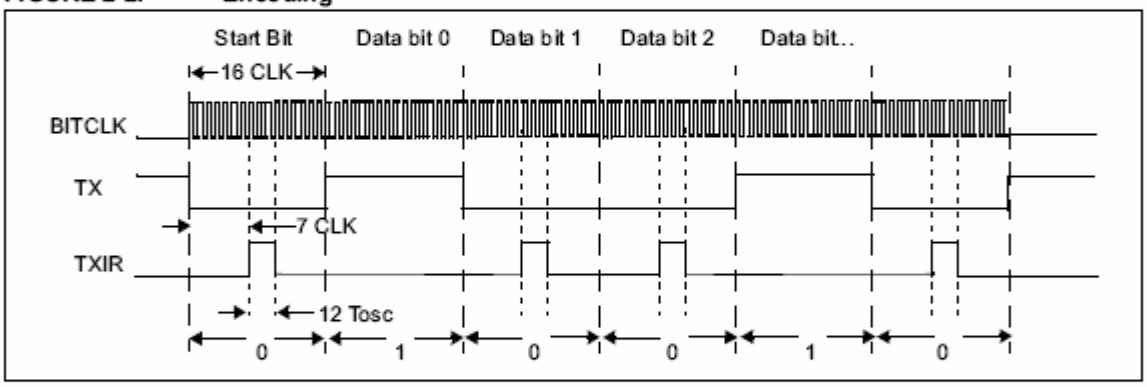
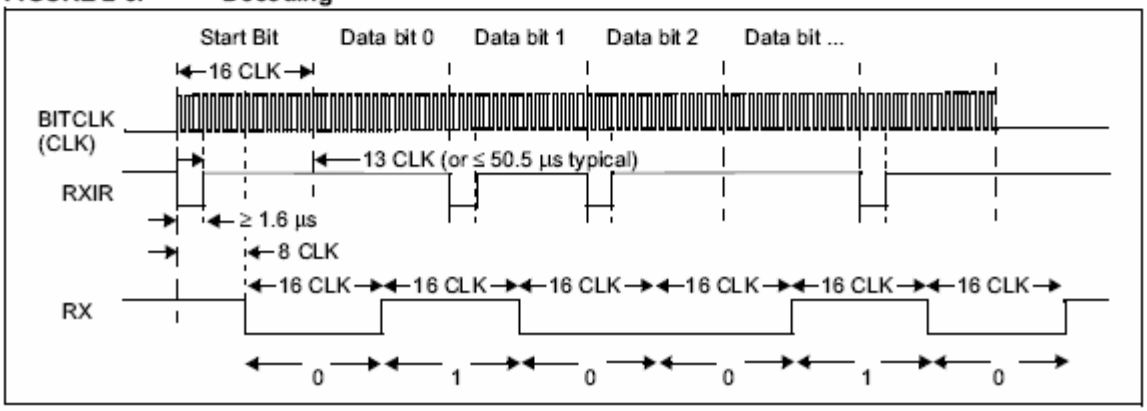


FIGURE 2-3: Decoding



2.7 Encoding/Decoding Jitter and Offset

Figure 2-4 shows the jitter and offset that is possible on the RX pin and the TXIR pin.

Jitter is the possible variation of the desired edge.

Offset is the propagation delay of the input signal (RXIR or TX) to the output signal (RX or TXIR).

The first bit on the output pin (on RX or TXIR) will show jitter compared to the input pin (RXIR or TX), but all remaining bits will be a constant distance.

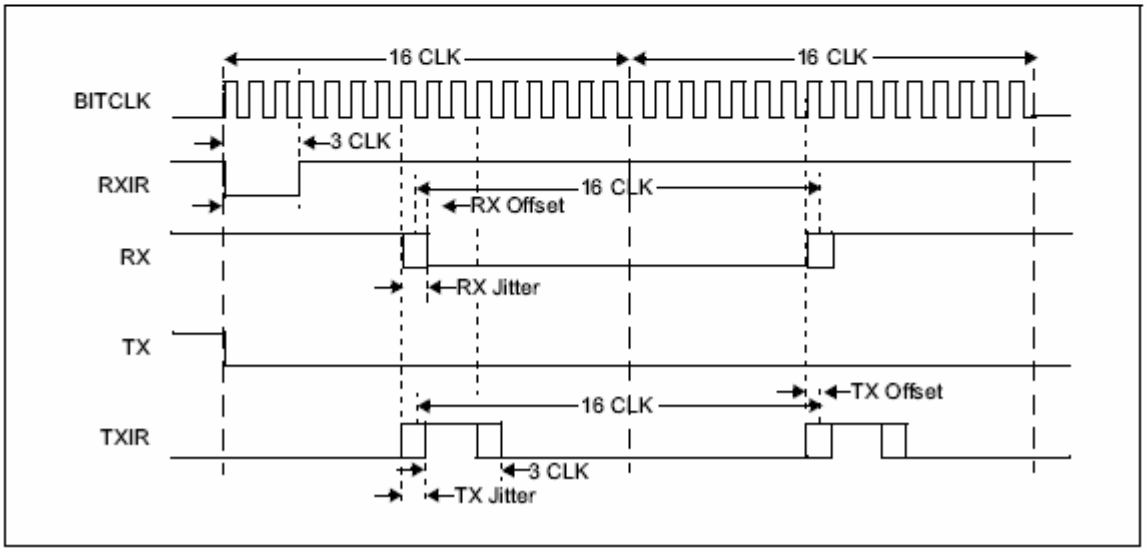
2.8 Minimizing Power

The device can be placed in a low power mode by disabling the device (holding the EN pin at the low state). The internal state machine is monitoring this pin for a low level, and once this is detected the device is disabled and enters into a low power state.

2.8.1 RETURNING TO OPERATION

When the device is disabled, the device is in a low power state. When the EN pin is brought to a high level, the device will return to the operating mode. The device requires a delay of 1000 TOSC before data may be transmitted or received.

FIGURE 2-4: Effects of Jitter and Offset



4.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings†

Ambient Temperature under bias ..... –40°C to +125°C

Storage Temperature ..... –65°C to +150°C

Voltage on V_{DD} with respect to V_{SS} ..... 0 to +7V

Voltage on $\overline{RESET}$ with respect to V_{SS} ..... 0 to +14V

Voltage on all other pins with respect to V_{SS} ..... –0.6V to (V_{DD} + 0.6V)

Total Power Dissipation (1) ..... 700 mW

Max. Current out of V_{SS} pin ..... 150 mA

Max. Current into V_{DD} pin ..... 125 mA

Input Clamp Current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ±20 mA

Output Clamp Current, I_{OK} (V_0 < 0 or V_0 > V_{DD}) ..... ±20 mA

Max. Output Current sunk by any Output pin ..... 25 mA

Max. Output Current sourced by any Output pin ..... 25 mA

Note 1: Power Dissipation is calculated as follows:

$$P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD}-V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$$

†NOTICE: Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

4.1 DC Characteristics

DC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial)				
Param. No.	Sym	Characteristic	Min	Typ(1)	Max	Units	Conditions
D001	VDD	Supply Voltage	2.5	—	5.5	V	See Figure 4-1
D002	VDR	RAM Data Retention Voltage (2)	2.5	—	—	V	Device Oscillator/Clock stopped
D003	VPOR	VDD Start Voltage to ensure Power-on Reset	—	VSS	—	V	
D004	SVDD	VDD Rise Rate to ensure Power-on Reset	0.05	—	—	V/ms	
D010	IDD	Supply Current (3)	— — — — — —	0.8 0.6 0.4 3 4 4.5	1.4 1.0 0.8 7 12 16	mA mA mA mA mA mA	FOSC = 4 MHz, VDD = 5.5V FOSC = 4 MHz, VDD = 3.0V FOSC = 4 MHz, VDD = 2.5V FOSC = 10 MHz, VDD = 3.0V FOSC = 20 MHz, VDD = 4.5V FOSC = 20 MHz, VDD = 5.5V
D020	IPD	Device Disabled Current (3, 4)	— — — —	0.25 0.25 0.4 3	4 3 5.5 8	µA µA µA µA	VDD = 3.0V, 0°C ≤ TA ≤ +70°C VDD = 2.5V, 0°C ≤ TA ≤ +70°C VDD = 4.5V, 0°C ≤ TA ≤ +70°C VDD = 5.5V, –40°C ≤ TA ≤ +85°C

Note 1: Data in the Typical ("Typ") column is based on characterization results at 25°C. This data is for design guidance only and is not tested.

2: This is the limit to which VDD can be lowered without losing RAM data.

3: The supply current is mainly a function of the operating voltage and frequency. Pin loading, pin rate, and temperature have an impact on the current consumption.

a) The test conditions for all IDD measurements are made when device is enabled (EN pin is high): OSC1 = external square wave, from rail-to-rail; all input pins pulled to VSS, RXIR = VDD, $\overline{\text{RESET}}$ = VDD;

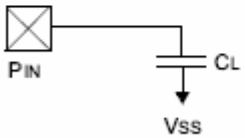
b) When device is disabled (EN pin is low), the conditions for current measurements are the same.

4: When the device is disabled (EN pin is low), current is measured with all input pins tied to VDD or VSS and the output pins driving a high or low level into infinite impedance.

TABLE 4-2: AC TEMPERATURE AND VOLTAGE SPECIFICATIONS

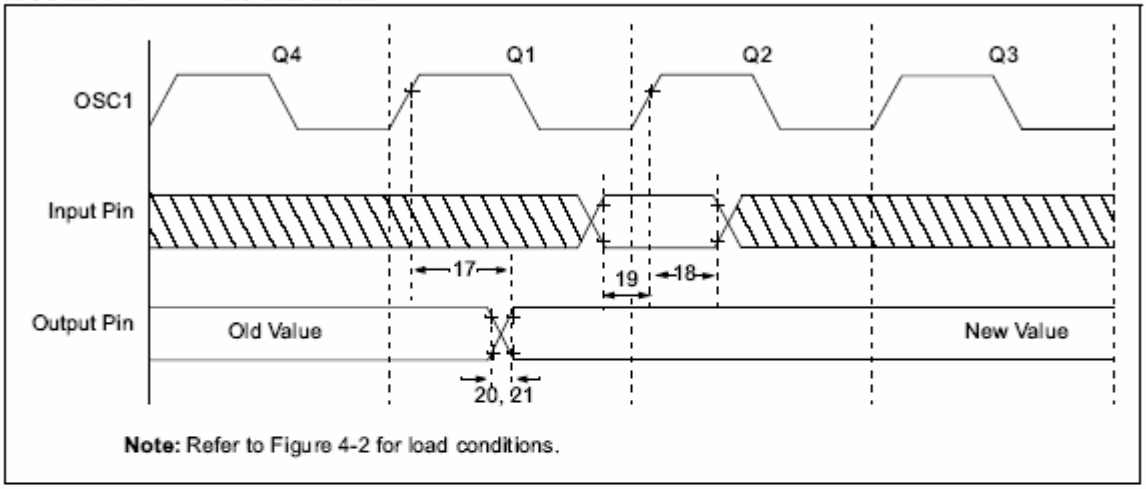
AC CHARACTERISTICS	Standard Operating Conditions (unless otherwise stated) Operating temperature –40°C ≤ TA ≤ +85°C (industrial) Operating voltage VDD range as described in DC spec Section 4.1.

FIGURE 4-2: Load Conditions for Device Timing Specifications



CL = 50 pF for all pins except OSC2
15 pF for OSC2 when external clock is used to drive OSC1

FIGURE 4-4: I/O Waveform



Note: Refer to Figure 4-2 for load conditions.

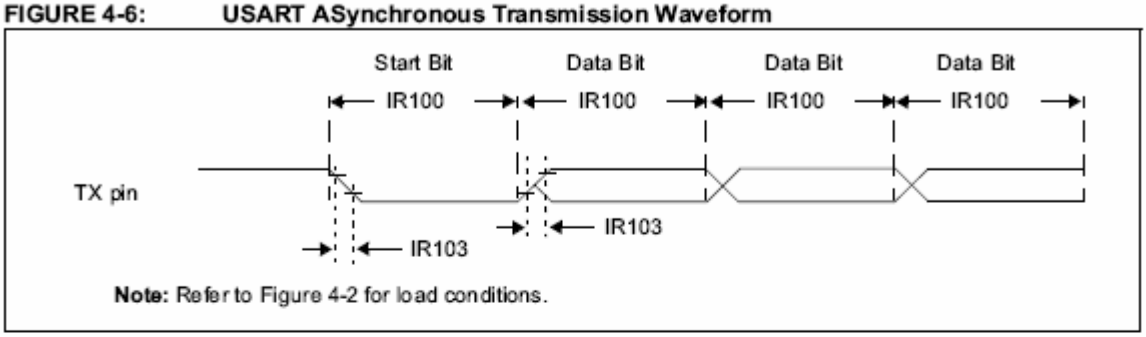
TABLE 4-4: I/O TIMING REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	Sym	Characteristic	Min	Typ(1)	Max	Units	Conditions
17	TosH2ioV	OSC1↑ (Q1 cycle) to Output valid (2)	—	—	100	ns	
18	TosH2ioI	OSC1↑ (Q2 cycle) to Input invalid (I/O in hold time)	200	—	—	ns	
19	TioV2osH	Input valid to OSC1↑ (I/O in setup time)	0	—	—	ns	
20	ToR	RX and TXIR pin rise time (2)	—	10	25	ns	
21	ToF	RX and TXIR pin fall time (2)	—	10	25	ns	

Note 1: Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated.

2: See Figure 4-2 for loading conditions.

FIGURE 4-6: USART ASynchronous Transmission Waveform



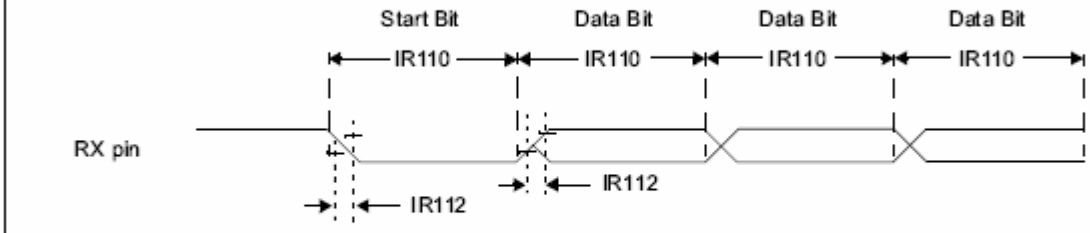
Note: Refer to Figure 4-2 for load conditions.

TABLE 4-6: USART ASYNCHRONOUS TRANSMISSION REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
IR100	TTXBIT	Transmit Baud rate					Hardware Selection
			768	—	768	TOSC	BAUD2:BAUD0 = 000
			384	—	384	TOSC	BAUD2:BAUD0 = 001
			192	—	192	TOSC	BAUD2:BAUD0 = 010
			128	—	128	TOSC	BAUD2:BAUD0 = 011
			64	—	64	TOSC	BAUD2:BAUD0 = 100
							Software Selection BAUD2:BAUD0 = 111
			768	—	768	TOSC	Hex Command = 0x87
			384	—	384	TOSC	Hex Command = 0x8B
			192	—	192	TOSC	Hex Command = 0x85
			128	—	128	TOSC	Hex Command = 0x83
			64	—	64	TOSC	Hex Command = 0x81
IR101	ETXBIT	Transmit (TX pin) Baud rate Error (into MCP2120)	—	—	1	%	
IR102	ETXIRBIT	Transmit (TXIR pin) Baud rate Error (out of MCP2120)(1)	—	—	1	%	
IR103	TTXRF	TX pin rise time and fall time	—	—	25	ns	

Note 1: This error is not additive to IR101 parameter.

FIGURE 4-7: USART ASynchronous Receive Timing



Note: Refer to Figure 4-2 for load conditions.

TABLE 4-7: USART ASYNCHRONOUS RECEIVE REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
IR110	TRXBIT	Receive Baud Rate					Hardware Selection
			768	—	768	TOSC	BAUD2:BAUD0 = 000
			384	—	384	TOSC	BAUD2:BAUD0 = 001
			192	—	192	TOSC	BAUD2:BAUD0 = 010
			128	—	128	TOSC	BAUD2:BAUD0 = 011
			64	—	64	TOSC	BAUD2:BAUD0 = 100
							Software Selection BAUD2:BAUD0 = 111
			768	—	768	TOSC	Hex Command = 0x87
			384	—	384	TOSC	Hex Command = 0x8B
			192	—	192	TOSC	Hex Command = 0x85
			128	—	128	TOSC	Hex Command = 0x83
			64	—	64	TOSC	Hex Command = 0x81
IR111	ERXBIT	Receive (RXIR pin) Baud rate Error (into MCP2120)	—	—	1	%	
IR112	ERXBIT	Receive (RX pin) Baud rate Error (out of MCP2120)(1)	—	—	1	%	
IR113	TTXRF	RX pin rise time and fall time	—	—	25	ns	

Note 1: This error is not additive to IR111 parameter.

FIGURE 4-8: TX and TXIR Waveforms

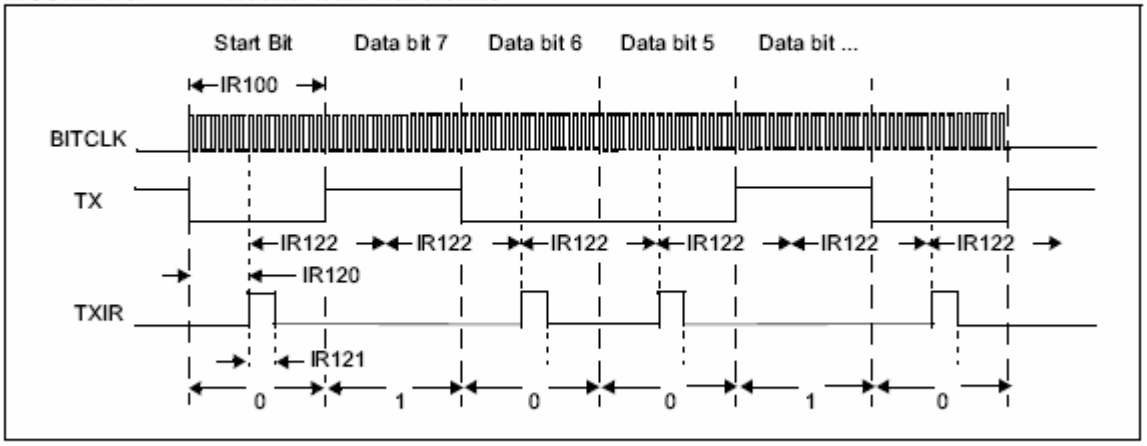


TABLE 4-8: TX AND TXIR REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
IR100	TTXBIT	Transmit Baud Rate					Hardware Selection
			768	—	768	TOSC	BAUD2:BAUD0 = 000
			384	—	384	TOSC	BAUD2:BAUD0 = 001
			192	—	192	TOSC	BAUD2:BAUD0 = 010
			128	—	128	TOSC	BAUD2:BAUD0 = 011
			64	—	64	TOSC	BAUD2:BAUD0 = 100
					8		Software Selection BAUD2:BAUD0 = 111
			768	—	768	TOSC	Hex Command = 0x87
			384	—	384	TOSC	Hex Command = 0x8B
			192	—	192	TOSC	Hex Command = 0x85
			128	—	128	TOSC	Hex Command = 0x83
			64	—	64	TOSC	Hex Command = 0x81
IR120	TTXL2TXIRH	TX falling edge (↓) to TXIR rising edge (↑) (1)	7TBITCLK - 8.34 µs	7	7TBITCLK + 8.34 µs	TBITCLK	
IR121	TTXIRPW	TXIR pulse width	12	—	12	TOSC	
IR122	TTXIRP	TXIR bit period (1)	—	16	—	TBITCLK	

Note 1: TBITCLK = TTXBIT/16

FIGURE 4-9: RXIR and RX Waveforms

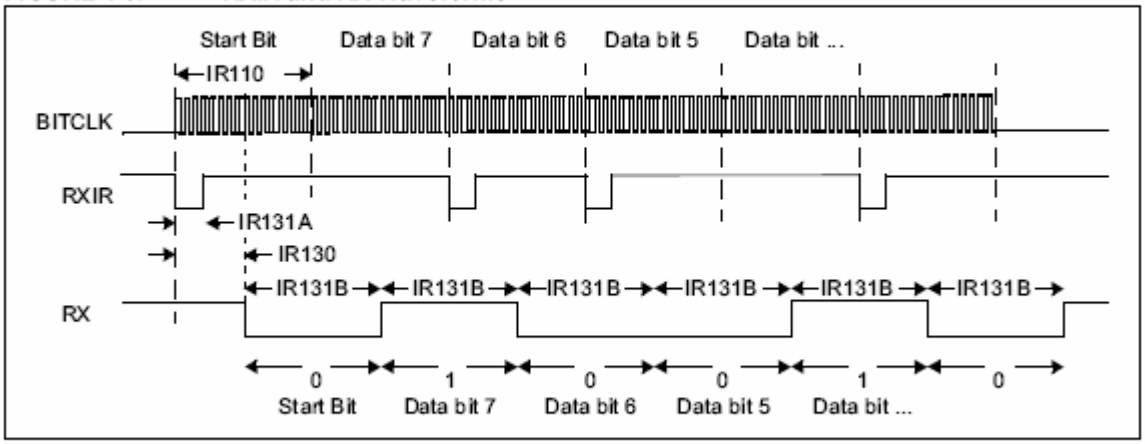


TABLE 4-9: RXIR REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
IR110	TRXBIT	Receive Baud Rate					Hardware Selection
			768	—	768	TOSC	BAUD2:BAUD0 = 000
			384	—	384	TOSC	BAUD2:BAUD0 = 001
			192	—	192	TOSC	BAUD2:BAUD0 = 010
			128	—	128	TOSC	BAUD2:BAUD0 = 011
			64	—	64	TOSC	BAUD2:BAUD0 = 100
							Software Selection BAUD2:BAUD0 = 111
			768	—	768	TOSC	Hex Command = 0x87
			384	—	384	TOSC	Hex Command = 0x8B
			192	—	192	TOSC	Hex Command = 0x85
			128	—	128	TOSC	Hex Command = 0x83
			64	—	64	TOSC	Hex Command = 0x81
IR130	TRXIRL2RXH	RXIR falling edge (↓) to RX falling edge (↓) (1)	8TBITCLK - 8.34 µs	8	8TBITCLK + 8.34 µs	TBITCLK	
IR131A	TRXIRPW	RXIR pulse width	3	—	3	TOSC	
IR132	TRXIRP	RXIR bit period (1)	—	16	—	TBITCLK	

Note 1: TBITCLK = TRXBIT/16

FIGURE 4-10: Command Mode: TX and RX Waveforms

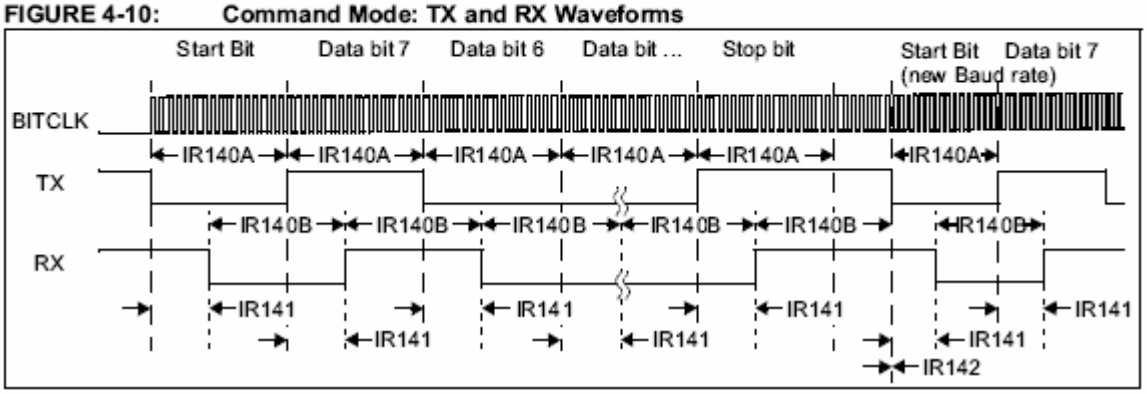


TABLE 4-10: TX AND TXIR REQUIREMENTS

AC Characteristics			Standard Operating Conditions (unless otherwise specified) Operating Temperature –40°C ≤ TA ≤ +85°C (industrial) Operating Voltage VDD range is described in Section 4.1				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
IR140A	BTX	Transmit Baud Rate	16	—	16	TBITCLK	
IR140B	BRX	Receive Baud Rate	16	—	16	TBITCLK	
IR141	TTXE2RXE	TX edge to RX edge (delay)	5.5	8	10.5	TBITCLK	
IR142	TRXP2TXS	Delay from RX Stop bit complete to TX Start bit (new baud rate)	—	—	0	TOSC	

APÉNDICE H. CÓDIGO MICROCONTROLADOR DE GESTIÓN Y CONTROL.

```
;*****************************************************************************************************************
;
;                                                   PROEX.ASM
;
;   DESARROLLO DE PROGRAMA CON EL MCU PIC16F877 PARA EL PROYECTO DE GRADUACIÓN:
;
;             "PROTOTIPO  EXPERIMENTAL  PARA LA ACTUALIZACION  DE  UNA BASE DE  DATOS
;              DESDE UN DISPOSITIVO MOVIL GSM ENFOCADO AL CAMPO DE LA DISTRIBUCION DE
;                                        HIDROCARBUROS LIQUIDOS."
;
;                       Integrantes:
;                                       Hernández Ayala, Francisco
;                                       Zepeda Cornejo, David Adrián
;                                       Pineda Martínez, Aldo Alberto
;
;*****************************************************************************************************************

                        LIST    p=16F877   ; Indica el modelo de PIC

                        #include "P16F877.INC"      ; Archivo de encabezado. Define el archivo
                                                    ; donde están definidos todos los registros
                                                    ; del PIC16F877A.

__CONFIG _CP_OFF & _WDT_OFF & _BODEN_ON & _PWRTE_ON & _XT_OSC & _LVP_OFF & _DEBUG_OFF &
_WRT_ENABLE_OFF

        ; Bits de configuración:
        ;               - Protección de Código:         Habilitada.
        ;               - WatchDog Timer:               Deshabilitado.
        ;               - Brown Out Reset:              Habilitado.
        ;               - PoWer-up Timer:               Habilitado.
        ;               - Tipo de Oscilador:            Cristal (HS).
        ;               - Low Voltage Program:          Habilitado.
        ;               - Background Debug:             Deshabilitado.


;*****************************************************************************************************************
;               1.      DECLARACIÓN DE VARIABLES.

;       VARIABLES DEL SISTEMA

STATUS          equ 0x03        ; Registro de Estado
FSR             equ 0x04        ; File Select Register
PORTA           equ 0x05        ; Registro de Datos Puerto A
PORTB           equ 0x06        ; Registro de Datos Puerto B
PORTC           equ 0x07        ; Registro de Datos Puerto C
PORTD           equ 0x08        ; Registro de Datos Puerto D
PORTE           equ 0x09        ; Registro de Datos Puerto E
PCLATH          equ 0x0A        ; Program Counter Latch High
INTCON          equ 0x0B        ; Registro de Control de Interrupción
PIR1            equ 0x0C        ; Registro de Banderas de Interrupción Periférica 1
PIR2            equ 0x0D        ; Registro de Banderas de Interrupción Periférica 2
TMR2            equ 0x11        ; Registro del Timer2
T2CON           equ 0x12        ; Registro de Cotrol del Timer2
```

```
ADRESH      equ 0x1E      ; Byte Superior del Registro de Resultado A/D
ADCON0      equ 0x1F      ; Registro de Control 0 A/D
TRISA       equ 0x85      ; Registro de Dirección de Datos Puerto A
TRISB       equ 0x86      ; Registro de Dirección de Datos Puerto B
TRISC       equ 0x87      ; Registro de Dirección de Datos Puerto C
TRISD       equ 0x88      ; Registro de Dirección de Datos Puerto D
PR2         equ 0x92      ; Registro de Período Timer2
ADRESL      equ 0x9E      ; Byte Inferior del Registro de Resultado A/D
ADCON1      equ 0x9F      ; Registro de Control 1 A/D
EEDATA      equ 0x10C     ; Byte Inferior del Registro de Datos EEPROM
EEADR       equ 0x10D     ; Byte Inferior del Registro de Dirección EEPROM
EEDATH      equ 0x10E     ; Byte Superior del Registro de Datos EEPROM
EEADRH      equ 0x10F     ; Byte Superior del Registro de Dirección EEPROM
EECON1      equ 0x18C     ; Registro de Control 1 EEPROM
EECON2      equ 0x18D     ; Registro de Control 2 EEPROM

C           equ 0         ; Bandera de STATUS, indica desborde de un resultado
DC          equ 1         ; Bandera de STATUS, indica desborde entre nibbles
Z           equ 2         ; Bandera de STATUS, indica que un resultado = 0
RP0         equ 5         ; Bit de STATUS, para Direccionamiento Directo de
                          ; Bancos de Memoria Bank0 a Bank3
RP1         equ 6         ; Bit de STATUS, para Direccionamiento Directo de
                          ; Bancos de Memoria Bank0 a Bank3
IRP         equ 7         ; Bit de STATUS, para Direccionamiento Indirecto de
                          ; Bancos de Memoria Bank0/Bank1 o Bank2/Bank3
w           equ 0         ; Para declarar como destino de operación a w
f           equ 1         ; Para declarar como destino de operación al mismo f
GO          equ 2         ; Bit de ADCON0, para iniciar una conversión A/D
ADIF        equ 6         ; Bandera de PIR1, indica fin de conversión A/D
TMR2ON      equ 2         ; Bit de T2CON, para iniciar marcha del Timer2
TMR2IF      equ 1         ; Bandera de PIR1, indica cuándo TMR2 alcanza a PR2
GIE         equ 7         ; Bandera de INTCON, para habilitar interrupciones
ADON        equ 0         ; Bit de ADCON0, para encender/apagar el ADC


;           VARIABLES DE USUARIO

ENT         equ 0x20      ; Entradas del Sistema
SAL         equ 0x21      ; Salidas del Sistema
CANAL       equ 0x22      ; Valor para Configurar y Eligir un Canal ADC
SUMAL       equ 0x23      ; Byte Inferior de la Suma de Resultados del ADC
SUMAH       equ 0x24      ; Byte Superior de la Suma de Resultados del ADC
SEG         equ 0x25      ; Contador de Segundos
PROML       equ 0x26      ; Byte Inferior del Valor Promedio de Conversiones
PROMH       equ 0x27      ; Byte Superior del Valor Promedio de Conversiones
LAPSO       equ 0x28      ; Variable a utilizar durante la espera de 1 segundo
VAR1        equ 0x29      ; Variable para propósito general
ALARMA      equ 0         ; Bit de VAR1 que indica una combinación prohibida

; Entradas al Sistema por Puerto B
B1          equ 0         ; Bit 0, para activar la Electroválvula1
B2          equ 1         ; Bit 1, para activar la Electroválvula2
B3          equ 2         ; Bit 2, para activar la Electroválvula3
LLAVE       equ 3         ; Bit 3, para llave de arranque del sistema
EPARO       equ 4         ; Bit 4, para el paro de sistema
REC         equ 5         ; Bit 5, indicación del PIC2 de PROM recibido
CEL         equ 6         ; Bit 6, señal de habilitación del sistema por el celular
```

```
; Salidas del Sistema por Puerto C
EV1             equ 0           ; Bit 0, para activar la Electroválvula1
EV2             equ 1           ; Bit 1, para activar la Electroválvula2
EV3             equ 2           ; Bit 2, para activar la Electroválvula3
MARCHA          equ 3           ; Bit 3, para activar LED indicador de Marcha
SPARO           equ 4           ; Bit 4, para activar LED indicador de Paro
ENV             equ 5           ; Bit 5, indicación al PIC2 de PROM enviado
ADH0            equ 6           ; Bit 6, para envío del bit 0 de PROMH
ADH1            equ 7           ; Bit 7, para envío del bit 1 de PROMH

; Salidas del Sistema por Puerto A
ALERTA          equ 2           ; Bit 2, para LED "combinación de entrada prohibida"


;*****************************************************************************************************************
;                       2.      INICIALIZACIÓN DE VARIABLES.


                ORG     0               ; Directiva para indicar compilación absoluta. Además
                                        ; indica la localidad de memoria a partir de la cual
                                        ; será alojado el programa.
                CLRF    STATUS          ; Limpia STATUS, eligiendo así el banco0
                CLRW                    ; Se limpia w, quedando con el valor 0x00
                MOVWF   SUMAL           ; Se limpia la variable SUMAL
                MOVWF   SUMAH           ; Se limpia la variable SUMAH
                MOVWF   SEG             ; Se limpia la variable SEG
                MOVWF   PROML           ; Se limpia la variable PROML
                MOVWF   PROMH           ; Se limpia la variable PROMH
                MOVWF   LAPSO           ; Se limpia la variable LAPSO
                MOVWF   VAR1            ; Se limpia la variable VAR1


;*****************************************************************************************************************
;               3.      CONFIGURACIÓN DE ADC, TIMER Y PUERTOS DIGITALES

                CLRF    STATUS          ; Limpia Registro STATUS (Queda con el valor 0x00)
; ADC
                BSF     STATUS, RP0     ; Se elige el Banco1 de Memoria de Datos
                MOVLW   0X04            ; Con el valor 0x24 se  configura a RA0, RA1 y RA3
                MOVWF   ADCON1          ; como entradas AN, y las demás como digitales. El
                                        ; resultado del ADC se justifica a la izquierda.
; TIMER2
                MOVLW   0x96
                MOVWF   PR2             ; Se carga a PR2 con una cuenta de 0x96 (150)
                BCF     STATUS, RP0     ; Se elige el Banco0 de Memoria de Datos
                CLRF    TMR2            ; Se limpia el registro TMR2
                MOVLW   0x73            ; Con el valor 0x73 se configura al Timer2 para
                MOVWF   T2CON           ; Preescala = 1:16, Postescala = 1:15, y apagado.
                                        ; Así, el overflow se obtiene como:
                                        ; Tov = 4/18432000 Hz * 16 * 15 * 150 = 0.0078125 seg
                                        ; que al ser repetido 128 veces, se obtiene
                                        ; una cuenta de 1 segundo.

;Puertos Digitales
                CLRF    PORTA           ; Se inicializa el puerto A limpiando los latchs de datos de salida
                CLRF    PORTB           ; Se inicializa el puerto B limpiando los latchs de datos de salida
```

```
                CLRF    PORTC           ; Se inicializa el puerto C limpiando los latchs de datos de salida
                CLRF    PORTD           ; Se inicializa el puerto D limpiando los latchs de datos de salida
                BSF     STATUS, RP0     ; Se elige el Banco1 de Memoria de Datos
                BCF     TRISA, ALERTA   ; Se establece el pin2 del puerto A como salida
                MOVLW   0xFF
                MOVWF   TRISB           ; Se establece todo el puerto B como entrada
                CLRF    TRISC           ; Se establece todo el puerto C como salida
                CLRF    TRISD           ; Se establece todo el puerto D como salida

;Interrupciones
                BCF     INTCON, GIE     ; Se desabilitan TODAS las interrupciones


;*******************************************************************************************************************
;               4.      ARRANQUE DEL SISTEMA Y PROCESAMIENTO DE ENTRADAS

                BCF     STATUS, RP0             ; Se elige el banco 0 de Memoria de datos
HAB             BTFSS   PORTB, CEL              ; ¿El celular ha habilitado al sistema?
                GOTO    HAB                     ; NO, esperar por habilitación de celular
                                                ; SI,
INICIO          BTFSS   PORTB,  LLAVE           ; ¿El sistema ha sido habilitado por la llave?
                GOTO    INICIO                  ; NO, esperar por llave de arranque
                BSF     PORTC, MARCHA           ; SÍ, activar LED indicador de Marcha

REGRESO         MOVF    PORTB, w                ; Leer el puerto de entrada
                MOVWF   ENT                     ; Almacenar la lectura de entradas en memoria
                BTFSC   ENT, EPARO              ; ¿Hay una solicitud de paro?
                BSF     PORTC, SPARO            ; SÍ, Activar led Paro
                BTFSC   ENT, EPARO              ; Y
                GOTO    REGRESO                 ; Regresar a leer las entradas
                BCF     PORTC, SPARO            ; NO, desactivar led PARO (si no lo está)

                MOVF    ENT, w                  ; Cargar nuevamente la lectura de entradas a w
                ANDLW   0X07                    ; Enmascara los bits de petición para activar EV's
                SUBLW   0X03
                BTFSC   STATUS, Z               ; ¿Se presionaron B1 y B2 simultáneamente?
                BSF     VAR1, ALARMA            ; SÍ, activar bandera ALARMA
                                                ; NO,

                MOVF    ENT, w                  ; Cargar nuevamente la lectura de entradas a w
                ANDLW   0X07                    ; Enmascara los bits de petición para activar EV's
                SUBLW   0X05
                BTFSC   STATUS, Z               ; ¿Se presionaron B1 y B3 simultáneamente?
                BSF     VAR1, ALARMA            ; SÍ, activar bandera ALARMA
                                                ; NO,

                MOVF    ENT, w                  ; Cargar nuevamente la lectura de entradas a w
                ANDLW   0X07                    ; Enmascara los bits de petición para activar EV's
                SUBLW   0X06
                BTFSC   STATUS, Z               ; ¿Se presionaron B2 y B3 simultáneamente?
                BSF     VAR1, ALARMA            ; SÍ, activar bandera ALARMA
                                                ; NO,

                MOVF    ENT, w                  ; Cargar nuevamente la lectura de entradas a w
                ANDLW   0X07                    ; Enmascara los bits de petición para activar EV's
                SUBLW   0X07
                BTFSC   STATUS, Z               ; ¿Se presionaron B1, B2 y B3 simultáneamente?
```

```
                BSF     VAR1, ALARMA            ; SÍ, activar bandera ALARMA
                                                ; NO,
                                                ; Entonces:
                BTFSS   VAR1, ALARMA            ; ¿Se ha detectado una combinación prohibida?
                GOTO    SALTO1                  ; NO, Continuar
                BSF     PORTA, ALERTA           ; SÍ, Activar led ALERTA
                CALL    SEGUNDO                 ; Subrutina de espera de 1 segundo
                BCF     T2CON, TMR2ON           ; Desabilita el Timer2
                CALL     SEGUNDO                ; Subrutina de espera de 1 segundo
                BCF     T2CON, TMR2ON           ; Desabilita el Timer2
                CALL     SEGUNDO                ; Subrutina de espera de 1 segundo
                BCF     T2CON, TMR2ON           ; Desabilita el Timer2
                CALL     SEGUNDO                ; Subrutina de espera de 1 segundo
                BCF     T2CON, TMR2ON           ; Desabilita el Timer2
                CALL     SEGUNDO                ; Subrutina de espera de 1 segundo
                BCF     T2CON, TMR2ON           ; Desabilita el Timer2
                CLRF    VAR1
                BCF     PORTA, ALERTA           ; Desactivar led ALERTA
                GOTO     REGRESO                ; Regresa a leer las entradas

SALTO1          CLRF    VAR1                    ; Limpia VAR1
                BTFSC   PORTB, LLAVE            ; ¿El sistema aún sigue habilitado?
                GOTO    SALTO2                  ; SÍ, continuar
                BCF     PORTC, MARCHA           ; NO, apagar LED indicador de Marcha
                GOTO    HAB                     ; Y Regresar a inicio.

SALTO2          BTFSS   ENT, B1                 ; ¿Se pide activar la EV1?
                GOTO    VERIF2                  ; NO, ir a verificar B2
                BSF     VAR1, EV1               ; SÍ, activar el bit para EV1 en VAR1
                MOVLW   0x81                    ; Valor para encender el ADC y
                MOVWF   CANAL                   ; elegir el canal 0 ADC
                GOTO    DES                     ; Salto a la etapa de DESCARGA

VERIF2          BTFSS   ENT, B2                 ; ¿Se pide activar la EV2?
                GOTO    VERIF3                  ; NO, ir a verificar B3
                BSF     VAR1, EV2               ; SÍ, activar el bit para EV2 en VAR1
                MOVLW   0x89                    ; Valor para encender el ADC y
                MOVWF   CANAL                   ; elegir el canal 1 ADC
                GOTO    DES                     ; Salto a la etapa de DESCARGA

VERIF3          BTFSS   ENT, B3                 ; ¿Se pide activar la EV3?
                GOTO    REGRESO                 ; NO, regresa a leer las entradas
                BSF     VAR1, EV3               ; SÍ, activar el bit para EV3 en VAR1
                MOVLW   0x99                    ; Valor para encender el ADC y
                MOVWF   CANAL                   ; elegir el canal 3 ADC
                                                ; Continúa a la etapa siguiente


;***************************************************************************************************************************
;                 5.  ACTIVACIÓN Y CONTROL DE SALIDAS (DESCARGA DE COMBUSTIBLE).
;                                    CAPTURA DE SEÑAL DE SENSOR.


DES             BTFSC   PORTB, LLAVE            ; ¿El sistema aún sigue habilitado?
                GOTO    SALTO3                  ; SÍ, continuar
                BCF     PORTC, MARCHA           ; NO, apagar LED indicador de Marcha
                CLRF    VAR1                    ; Se limpia la variable VAR1
```

```
                GOTO    HAB             ; Y Regresar a inicio.

SALTO3          BTFSS   PORTB,  EPARO   ; ¿Hay una solicitud de paro?
                GOTO    SALTO4          ; NO, continuar
                BSF     PORTC, SPARO    ; SÍ, activar LED indicador de Paro
                CLRF    VAR1            ; Se limpia la variable VAR1
                GOTO    REGRESO         ; Regresa a leer las entradas

SALTO4          MOVLW   0x08            ; Se activa el bit de la electroválvula
                ADDWF   VAR1, w         ; correspondiente en el puerto C, sin
                MOVWF   PORTC           ; desactivar el led de marcha (bit 7)

RETRO           CALL    SEGUNDO         ; Llama la rutina de espera de 1 seg. Luego de esta, el
                                        ;Timer2 seguirá corriendo.

                MOVF    CANAL, w        ; Se activa el ADC y se elige el canal
                MOVWF   ADCON0          ; según el contenido del registro CANAL

                                        ; Retardo de tiempo para el Tacq.
                MOVLW   0XC5            ; Tiempo de adquisición que necesita el ADC
TACQ            ADDLW   1               ; para realizar el muestreo del valor
                BTFSS   STATUS, Z       ; analógico presente en su entrada, en base
                GOTO    TACQ            ; al cual devolverá un valor digital
                                        ; Por seguridad, este dura aprox. 50useg
```

;Ya que la descarga sigue en progreso, sería deseable que el conteo de tiempo sea lo más aproximado a la realidad posible. Con este propósito, se deja correr el Timer2 sin interrumpirlo. Para que ocurra un overflow (TMR2IF=1)
;sería necesario que se ejecutaran 18,000 instrucciones, de 2Tcy cada una. Por lo tanto, se descarta el riesgo de que ocurra un overflow mientras el programa continúa.

```
                BSF     ADCON0, GO      ; Inicia la conversión ADC

CONVER          BTFSS   PIR1, ADIF      ; ¿Conversión hecha?
                GOTO    CONVER          ; NO, continuar esperando
                BCF     PIR1, ADIF      ; SÍ, limpiar bandera de overflow
                BCF     ADCON0, ADON    ; Apaga el módulo ADC

                MOVF    ADRESH, w       ; Carga uno de los bytes del resultado ADC
                MOVWF   PROMH           ; y lo almacena en memoria
                BSF     STATUS, RP0     ; Cambia a Banco0
                MOVF    ADRESL, w       ; Carga el otro byte del resultado ADC
                BCF     STATUS, RP0     ; y lo almacena en memoria
                MOVWF   PROML           ;

                BCF     STATUS, C       ; Elabora una serie de rotaciones entre
                MOVLW   0x06            ; los bytes de resultado ADC, de tal forma
                MOVWF   VAR1            ; que en PROML se alojen los 8 bits menos
ROTAR           RRF     PROMH, f        ; significativos, y en PROMH los 2 bits más
                RRF     PROML, f        ; significativos.
                DECFSZ  VAR1, f         ;
                GOTO    ROTAR           ;

                MOVF    PROML, w        ; Se carga en w el BMS de la respuesta ADC
                ADDWF   SUMAL, f        ; Acumula el valor devuelto por el ADC junto
                BTFSC   STATUS, C       ; con los valores de las conversiones
                INCF    SUMAH, f        ; anteriores en la variable SUMAH (parte alta)
```

```
                MOVF    PROMH, w        ; y SUMAL (parte baja), tomando en cuenta si
                ADDWF   SUMAH, f        ; ocurrió un desbordamiento.
                CLRF    PROMH
                CLRF    PROML

                INCF    SEG, f          ; Incrementa contador de segundos

                MOVF    SEG, w          ; Carga en w la cuenta actual de segundos
                SUBLW   0x3C            ; Resta 60 al contenido de SEG
                BTFSC   STATUS, Z       ; ¿La cuenta llegó a 60 segundos? (¿SEG=60?)
                CALL    PROM            ; SÍ, calcular y enviar el promedio de resultados ADC
                                        ; NO, continuar

;Verificación de Petición de Paro o Apagado del Sistema
                MOVF    PORTB, w        ; Se lee el puerto de entrada
                MOVWF   ENT             ; y se almacena la lectura en ENT
                BTFSC   ENT, LLAVE      ; ¿El sistema aún sigue habilitado?
                GOTO    SALTO5          ; SÍ, continuar
                CLRF    PORTC           ; NO, limpiar todas las salidas
                CLRF    PORTD           ;
                GOTO    SALTO6

SALTO5  BTFSS ENT, EPARO                ; ¿Hay una solicitud de paro?
                GOTO    RETRO           ; NO, regresar a esperar otro segundo
                CLRF    PORTD           ; SÍ, limpiar las salidas
                MOVLW   0x18            ; Desactivar cualquier led y electroválvula
                MOVWF PORTC             ; Sólo dejar activos los led PARO y MARCHA


SALTO6  MOVF SEG, w                     ; ¿El contador de segundos está limpio?, o sea
                BTFSC   STATUS, Z       ; ¿Se ha completado un proceso de 1 minuto?
                GOTO    SALTO7          ; SÍ, ir hacia SALTO7
                CALL    PROMT           ; NO, completar el proceso para ese lapso de
                                        ; tiempo inferior al minuto y enviarlo al PIC2

SALTO7  BCF     T2CON,  TMR2ON          ; Desabilita el Timer2
                CLRF    TMR2            ; Limpia el registro TMR2
                                        ; Ya que se ha limpiado el registro TMR2, es
                                        ; necesario reconfigurarlo

; Reconfiguración del TIMER2
                BSF     STATUS, RP0     ; Se elige el Banco1 de Memoria de Datos
                MOVLW   0x96
                MOVWF   PR2             ; Se carga a PR2 con una cuenta de 0xF9 (249)
                BCF     STATUS, RP0     ; Se elige el Banco0 de Memoria de Datos
                CLRF    TMR2            ; Se limpia el registro TMR2
                MOVLW   0x73            ; Con el valor 0x73 se configura al Timer2 para
                MOVWF   T2CON           ; Preescala = 1:16, Postescala = 1:15, y apagado.
                                        ; Así, el overflow se obtiene como:
                                        ; Tov = 4/18432000MHz * 16 * 15 * 150 = 0.0078125 seg
                                        ; que al ser repetido 128 veces, se obtiene
                                        ; una cuenta de 1 segundo.

                CLRF    SAL             ; Se limpian todas las variables utilizadas
                CLRF    CANAL           ; así como los puertos de salida.
                CLRF    SUMAL           ;
                CLRF    SUMAH           ;
```

```
                CLRF    SEG                     ;
                CLRF    PROML                   ;
                CLRF    PROMH                   ;
                CLRF    LAPSO                   ;
                CLRF    VAR1                    ;


                BTFSS   ENT, LLAVE              ; ¿El sistema ha sido detenido por la llave?
                GOTO    HAB                     ; SÍ, esperar por el arranque nuevamente
                GOTO    REGRESO                 ; NO, regresar a la espera por fin de paro

;*****************************************************************************************************************
;               SUBRUTINA PARA EL CÁLCULO Y ENVÍO DEL VALOR PROMEDIO
;               DE LOS RESULTADOS DEVUELTOS POR EL ADC EN 1 MINUTO

PROM            BCF     STATUS, RP0             ; Se elige el Banco0 de Memoria de Datos

;Cálculo del valor promedio de los resultados del ADC en 1 minuto.
CALC            MOVF    SEG, w                  ;
                SUBWF   SUMAL   , f             ;
                BTFSS   STATUS, C               ; ¿El resultado fué negativo?
                GOTO    NEG                     ; SÍ, ir a NEG.
INC             INCFSZ  PROML, f                ; NO, incrementar el valor del promedio.
                GOTO    CALC                    ; Y si no hubo carry, continuar el cálculo.
                INCF    PROMH, f                ; Pero si hubo carry, incrementar antes PROMH
                GOTO    CALC                    ; y luego continuar el cálculo.
NEG             MOVF    SUMAH, w                ;
                BTFSC   STATUS, Z               ; ¿El byte superior es 0?
                GOTO    ENVIO                   ; SÍ, Salir Y enviar el promedio al PIC2
                DECF    SUMAH, f                ; NO, decrementar el byte superior del prom
                GOTO    INC                     ; Continuar con el cálculo

;Envio del resultado promedio al PIC2.
ENVIO           MOVF    PROML, w                ; Coloca el byte menos significativo del
                MOVWF   PORTD                   ; promedio en el puerto D.
                BTFSC   PROMH, 0                ; ¿El bit 0 de PROMH es igual a 1?
                BSF     PORTC, ADH0             ; SÍ, activarlo en el puerto C

                NOP                             ; Ya que se ejecutan consecutivamente dos
                NOP                             ; comandos leer-modificar-escribir (BSF) en un
                NOP                             ; puerto, se han agregado este bloque de NOP
                NOP                             ; para que los comandos sean efectivos.

                BTFSC   PROMH, 1                ; ¿El bit 1 de PROMH es igual a 1?
                BSF     PORTC, ADH1             ; SÍ, activarlo en el puerto C

                NOP                             ; Ya que se ejecutan consecutivamente dos
                NOP                             ; comandos leer-modificar-escribir (BSF) en un
                NOP                             ; puerto, se han agregado este bloque de NOP
                NOP                             ; para que los comandos sean efectivos.

                BSF     PORTC, ENV              ; Indica al PIC2 que ya envió PROM

ESPIC2  BTFSS PORTB, REC                        ; ¿El PIC2 indica que ya capturó PROML y PROMH?
                GOTO    ESPIC2                  ; NO, Espera por la respuesta del PIC2
                                                ; SÍ,
                CLRF    PORTD                   ; Se limpian los puertos y bits de salida
```

```
                MOVLW  0x1F                 ; utilizados para enviar el promedio.
                ANDWF  PORTC, f             ;
                CLRF   SUMAL                ; Se limpia la suma de resultados del ADC
                CLRF   SUMAH                ;
                CLRF   PROML                ; Se limpia el resultado promedio calculado
                CLRF   PROMH                ;
                CLRF   SEG                  ; Se limpia el contador de segundos
                RETURN                      ; Fin de subrutina


;*****************************************************************************************************************************
;
;                       SUBRUTINA PARA EL CÁLCULO Y ENVÍO DEL VALOR PROMEDIO
;                        DE LOS RESULTADOS DEVUELTOS POR EL ADC PARA UN TIEMPO
;                                       INFERIOR A 1 MINUTO


PROMT           BCF    STATUS, RP0          ; Se elige el Banco0 de Memoria de Datos

;Cálculo del valor promedio de los resultados del ADC para menos de minuto.
CAL             MOVLW  0x3C                 ; Se carga el valor de 60
                SUBWF  SUMAL   , f          ;
                BTFSS  STATUS, C            ; ¿El resultado fué negativo?
                GOTO   NEGA                 ; SÍ, ir a neg.
CONC            INCFSZ PROML, f             ; NO, incrementar el valor del promedio.
                GOTO   CAL                  ; Y si no hubo desbordamiento, continuar el cálculo.
                INCF   PROMH, f             ; Pero si hubo desbordamiento, incrementar antes PROMH
                GOTO   CAL                  ; y luego continuar el cálculo.
NEGA            MOVF   SUMAH, w             ;
                BTFSC  STATUS, Z            ; ¿El byte superior es 0?
                GOTO   ENVIAR               ; SÍ, ir a enviar el promedio al PIC2
                DECF   SUMAH, f             ; NO, decrementar el byte superior del prom
                GOTO   CONC                 ; Continuar con el cálculo

;Envio del resultado promedio al PIC2.
ENVIAR          MOVF   PROML, w             ; Coloca el byte menos significativo del
                MOVWF  PORTD                ; promedio en el puerto D.
                BTFSC  PROMH, 0             ; ¿El bit 0 de PROMH es igual a 1?
                BSF    PORTC, ADH0          ; SÍ, activarlo en el puerto C

                NOP                         ; Ya que se ejecutan consecutivamente dos
                NOP                         ; comandos leer-modificar-escribir (BSF) en un
                NOP                         ; puerto, se han agregado este bloque de NOP
                NOP                         ; para que los comandos sean efectivos.

                BTFSC  PROMH, 1             ; ¿El bit 1 de PROMH es igual a 1?
                BSF    PORTC, ADH1          ; SÍ, activarlo en el puerto C

                NOP                         ; Ya que se ejecutan consecutivamente dos
                NOP                         ; comandos leer-modificar-escribir (BSF) en un
                NOP                         ; puerto, se han agregado este bloque de NOP
                NOP                         ; para que los comandos sean efectivos.

                BSF    PORTC, ENV           ; Indica al PIC2 que ya envió PROM

ESPREC          BTFSS  PORTB, REC           ; ¿El PIC2 indica que ya capturó PROML y PROMH?
                GOTO   ESPREC               ; NO, Espera por la respuesta del PIC2
                                            ; SÍ,
```

```
                CLRF    PORTD           ; Se limpian los puertos y bits de salida
                MOVLW   0x1F            ; utilizados para enviar el promedio.
                ANDWF   PORTC, f                ;
                CLRF    SUMAL           ; Se limpia la suma de resultados del ADC
                CLRF    SUMAH           ;
                CLRF    PROML           ; Se limpia el resultado promedio calculado
                CLRF    PROMH           ;
                CLRF    SEG             ; Se limpia el contador de segundos
                RETURN                  ; Fin de subrutina


;*****************************************************************************************************************
;               SUBRUTINA DE ESPERA PARA UN LAPSO DE TIEMPO DE 1 SEGUNDO


SEGUNDO         BCF     STATUS, RP0     ; Selecciona Banco0 de Memoria de Datos
                CLRF    LAPSO           ; Limpia la variable para espera de 1 seg
                BSF     T2CON, TMR2ON   ; Habilita el Timer2 e inicia la cuenta
ESPERA          BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    ESPERA          ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow

                INCF LAPSO, f           ; Incrementa el contador de overflows
                BTFSS LAPSO, 7          ; ¿La cuenta = 128 (tiempo = 1 seg)?
                GOTO ESPERA                     ; NO, continuar esperando
                CLRF    LAPSO           ; SÍ, Limpia la variable de uso general
                RETURN                          ; Fin de subrutina
```

;NOTA IMPORTANTE: Luego de la rutina, el Timer2 seguirá corriendo, por lo cual debe desactivarse abajo de la instrucción CALL si este ya no se ocupará más.

```
                END
```

APÉNDICE I. CÓDIGO MICROCONTROLADOR DE PROCESAMIENTO.

```
;*****************************************************************************************************************
;
;                                                PROEX2.ASM
;
;   DESARROLLO DE PROGRAMA CON EL MCU PIC16F877 PARA EL PROYECTO DE GRADUACIÓN:
;
;             "PROTOTIPO  EXPERIMENTAL  PARA LA ACTUALIZACION  DE  UNA BASE DE  DATOS
;              DESDE UN DISPOSITIVO MOVIL GSM ENFOCADO AL CAMPO DE LA DISTRIBUCION DE
;                                        HIDROCARBUROS LIQUIDOS."
;
;                                Integrantes:
;                                                Hernández Ayala, Francisco
;                                                Zepeda Cornejo,  David Adrián
;                                                Pineda Martínez, Aldo Alberto
;
;*****************************************************************************************************************



                                LIST    p=16F877   ; Indica el modelo de PIC

                                #include "P16F877.INC"      ; Archivo de encabezado. Define el archivo
                                                            ; donde están definidos todos los registros
                                                            ; del PIC16F877A.

__CONFIG _CP_OFF & _WDT_OFF & _BODEN_ON & _PWRTE_ON & _HS_OSC & _LVP_OFF & _DEBUG_OFF &
_WRT_ENABLE_OFF



        ; Bits de configuración:
                ;               - Protección de Código:         Habilitada.
                ;               - WatchDog Timer:               Deshabilitado.
                ;               - Brown Out Reset:              Habilitado.
                ;               - PoWer-up Timer:               Habilitado.
                ;               - Tipo de Oscilador:            Cristal (HS).
                ;               - Low Voltage Program:          Habilitado.
                ;               - Background Debug:             Deshabilitado.


;*****************************************************************************************************************

;                                       DECLARACIÓN DE VARIABLES.

;           VARIABLES DEL SISTEMA

STATUS          equ 0x03        ; Registro de Estado
FSR             equ 0x04        ; File Select Register
PORTA           equ 0x05        ; Registro de Datos Puerto A
PORTB           equ 0x06        ; Registro de Datos Puerto B
PORTC           equ 0x07        ; Registro de Datos Puerto C
```

```
PORTD       equ 0x08        ; Registro de Datos Puerto D
PORTE       equ 0x09        ; Registro de Datos Puerto E
PCLATH      equ 0x0A        ; Program Counter Latch High
INTCON      equ 0x0B        ; Registro de Control de Interrupción
PIR1        equ 0x0c        ; Registro de Banderas de Interrupción Periférica 1
PIR2        equ 0x0D        ; Registro de Banderas de Interrupción Periférica 2
TMR2        equ 0x11        ; Registro del Timer2
T2CON       equ 0x12        ; Registro de Cotrol del Timer2
RCSTA       equ 0x18        ; Registro de Control y Estado de Recepción USART
TXREG       equ 0x19        ; Registro para Transmisión de Datos USART
RCREG       equ 0x1A        ; Registro para Recepción de Datos USART
ADRESH      equ 0x1E        ; Byte Superior del Registro de Resultado A/D
ADCON0      equ 0x1F        ; Registro de Control 0 A/D
TRISA       equ 0x85        ; Registro de Dirección de Datos Puerto A
TRISB       equ 0x86        ; Registro de Dirección de Datos Puerto B
TRISC       equ 0x87        ; Registro de Dirección de Datos Puerto C
TRISD       equ 0x88        ; Registro de Dirección de Datos Puerto D
PR2         equ 0x92        ; Registro de Período Timer2
TXSTA       equ 0x98        ; Registro de Control y Estado de Tranmisión USART
SPBRG       equ 0x99        ; Registro de Período para el módulo USART
ADRESL      equ 0x9E        ; Byte Inferior del Registro de Resultado A/D
ADCON1      equ 0x9F        ; Registro de Control 1 A/D
EEDATA      equ 0x10C       ; Byte Inferior del Registro de Datos EEPROM
EEADR       equ 0x10D       ; Byte Inferior del Registro de Dirección EEPROM
EEDATH      equ 0x10E       ; Byte Superior del Registro de Datos EEPROM
EEADRH      equ 0x10F       ; Byte Superior del Registro de Dirección EEPROM
EECON1      equ 0x18C       ; Registro de Control 1 EEPROM
EECON2      equ 0x18D       ; Registro de Control 2 EEPROM


C           equ 0   ; Bandera de STATUS, indica desborde de un resultado
DC          equ 1   ; Bandera de STATUS, indica desborde entre nibbles
Z           equ 2   ; Bandera de STATUS, indica que un resultado = 0
RP0         equ 5   ; Bit de STATUS, para Direccionamiento Directo de
                    ; Bancos de Memoria Bank0 a Bank3
RP1         equ 6   ; Bit de STATUS, para Direccionamiento Directo de
                    ; Bancos de Memoria Bank0 a Bank3
IRP         equ 7   ; Bit de STATUS, para Direccionamiento Indirecto de
                    ; Bancos de Memoria Bank0/Bank1 o Bank2/Bank3
w           equ 0   ; Para declarar como destino de operación a w
f           equ 1   ; Para declarar como destino de operación al mismo f
GO          equ 2   ; Bit de ADCON0, para iniciar una conversión A/D
ADIF        equ 6   ; Bandera de PIR1, indica fin de conversión A/D
TMR2ON      equ 2   ; Bit de T2CON, para iniciar marcha del Timer2
TMR2IF      equ 1   ; Bandera de PIR1, indica cuándo TMR2 alcanza a PR2
GIE         equ 7   ; Bandera de INTCON, para habilitar interrupciones
ADON        equ 0   ; Bit de ADCON0, para encender/apagar el ADC
TXIF        equ 4   ; Bit de PIR1, indica que el buffer TXREG del USART está vacío.
RCIF        equ 5   ; Bit de PIR1, indica que el buffer RCREG del USART está lleno.
TRMT        equ 1   ; Bit de TXSTA, indica transmisión USART finalizada.
SPEN        equ 7   ; Bit de RCSTA, selecciona los pines RX y TX para el puerto serial.
OERR        equ 1   ; Bit de RCSTA, indicador de error Overrun del puerto de entrada USART.
CREN        equ 4   ; Bit de RCSTA, para habilitar recepción contínua

;           VARIABLES DE USUARIO
```

```
GAS             equ 0x20        ; Valor que identifica el tipo de combustible
TOTAL1H         equ 0x21        ; Byte superior del la suma de los promedios de conversiones de EV1
TOTAL1L         equ 0x22        ; Byte inferior del la suma de los promedios de conversiones de EV1
TOTAL2H         equ 0x23        ; Byte superior del la suma de los promedios de conversiones de EV2
TOTAL2L         equ 0x24        ; Byte inferior del la suma de los promedios de conversiones de EV2
TOTAL3H         equ 0x25        ; Byte superior del la suma de los promedios de conversiones de EV3
TOTAL3L         equ 0x26        ; Byte inferior del la suma de los promedios de conversiones de EV3
VAR             equ 0x27        ; Variable de propósito general

; Entradas al Sistema por Puerto A
EV1     equ 0           ; Bit 0, para activar la Electroválvula1
EV2     equ 1           ; Bit 1, para activar la Electroválvula2
EV3     equ 2           ; Bit 2, para activar la Electroválvula3
LLAVE   equ 3           ; Bit 3, para llave de arranque del sistema
ENV     equ 5           ; Bit 5, indicación del PIC1 de PROM enviado

; Salidas del sistema por Puerto C
IRDA    equ 3           ; Bit 3, para configurar transceptor IRDA

; Entradas al sistema por Puerto D
ADH0    equ 0           ; Bit 0, para recibir el bit 0 de PROMH
ADH1    equ 1           ; Bit 1, para recobor el bit 1 de PROMH

; Salidas del Sistema por Puerto D
REC     equ 5           ; Bit 5, indicación al PIC1 de PROM recibido
CEL     equ 6           ; Bit 6, indicación al PIC1 de habilitación por el celular.




;*****************************************************************************************************************************************
;               1.      CONFIGURACIÓN DE PUERTOS DIGITALES Y MODULO USART


                ORG     0       ; Directiva para indicar compilación absoluta. Además
                                ; indica la localidad de memoria a partir de la cual
                                ; será alojado el programa.

                CLRF    STATUS          ; Limpia Registro STATUS (Queda con el valor 0x00)


; Puertos Digitales
                CLRF    PORTA           ; Se inicializa el puerto A limpiando los latchs de datos de salida
                CLRF    PORTB           ; Se inicializa el puerto B limpiando los latchs de datos de salida
                CLRF    PORTC           ; Se inicializa el puerto C limpiando los latchs de datos de salida
                CLRF    PORTD           ; Se inicializa el puerto D limpiando los latchs de datos de salida

                BSF     STATUS, RP0     ; Se elige el Banco1 de Memoria de Datos
                MOVLW   0x07            ; Configuración necesaria en ADCON1 del ADC para
                MOVWF   ADCON1          ; permitir que el puerto A sea utilizado
                MOVLW   0xFF
                MOVWF   TRISA           ; Se establece el puerto A como entrada (RA5-RA0)
                MOVWF   TRISB           ; Se establece todo el puerto B como entrada
                MOVWF   TRISC           ; Se establece todo el puerto C como entrada
                MOVLW   0x07            ;
                MOVWF   TRISD           ; Se establecen los pines RD7-RD3 como salidas
```

```
                                        ; y los pines RD2-RD0 como entradas.


                BSF     STATUS, RP0     ; Se elige el Banco1 de Memoria de Datos
                MOVLW   0xFF
                MOVWF   TRISC           ; Se establece todo el puerto C como entrada

; Módulo USART
                MOVLW   0x1d            ; Valor en el registro SPBRG para
                MOVWF   SPBRG           ; establecer el Baud Rate deseado
                MOVLW   0x20            ; Configura el transmisor con 8 bits de datos,
                MOVWF   TXSTA           ; modo asíncrono, alta velocidad y habilitado.
                BCF     STATUS, RP0     ; Selección de Banco 0 de Memoria
                MOVLW   0x90            ; Configura el receptor con 8 bits de datos,
                MOVWF   RCSTA           ; y habilitado.Configura los pines RX y TX como
                                        ; pines utilizados para el puerto serial

; TIMER2
                BSF     STATUS, RP0     ; Selección de Banco 1 de Memoria
                MOVLW   0XC0
                MOVWF   PR2             ; Se carga a PR2 con una cuenta de 0xC0 (192)
                BCF     STATUS, RP0     ; Se elige el Banco0 de Memoria de Datos
                CLRF    TMR2            ; Se limpia el registro TMR2
                MOVLW   0x73            ; Con el valor 0x73 se configura al Timer2 para
                MOVWF   T2CON           ; Preescala = 1:16, Postescala = 1:15, y apagado.
                                        ; Así, el overflow se obtiene como:
                                        ; Tov = 4/18432000Hz * 16 * 15 * 192 = 100 mseg

                BCF     STATUS, RP0     ; Selección de Banco0 de Memoria


; Interrupciones
                BCF     INTCON, GIE     ; Se desabilitan TODAS las interrupciones




;*****************************************************************************************************************************
;               2.      ARRANQUE DEL SISTEMA Y CAPTURA DE DATOS DEL PIC1

                CLRF    STATUS          ; Limpia STATUS para elegir Banco0 de Memoria.
                CLRF    TOTAL1H         ; Se limpian las variables que almacenan los totales
                CLRF    TOTAL1L         ;
                CLRF    TOTAL2H         ;
                CLRF    TOTAL2L         ;
                CLRF    TOTAL3H         ;
                CLRF    TOTAL3L         ;

INICIO          CALL    LIMPIAR         ; Llama a la subrutina de limpieza del área GPR

; Inicialización del Registro de Direccionamiento Indirecto, FSR.
                CLRF    STATUS          ; Limpia STATUS para elegir Banco0 de Memoria.
                                        ; Además, hace que IRP=0 (Banco0/Banco1)
                MOVLW   0x28            ; 1ª dirección del área GPR que se traslada
```

```
                MOVWF FSR               ; al Registro de Direccionamiento Indirecto
                                        ; para almacenar los datos deseados (tabla).

; Espera a que el celular transmita un dato para habilitar el sistema.
HAB             BTFSS   PIR1, RCIF      ; ¿Se ha recibido un dato en el puerto USART?
                GOTO    HAB             ; NO, seguir esperando por otro dato.
                MOVF    RCREG, w        ; SÍ, leer el dato

                BSF     PORTD, CEL      ; Indica al PIC1 que el teléfono habilitó al sistema


ESPERA BTFSC    PORTA, LLAVE    ; ¿El sistema ha sido habilitado por la llave?
                GOTO    DESACT          ; SÍ, continuar
                BTFSC   PIR1, RCIF      ; ¿Se ha recibido un dato en el puerto USART?
                MOVF    RCREG, w        ; SÍ, leer el buffer de entrada para evitar un posible
                                        ; desbordamiento
                BTFSS   RCSTA, OERR     ; ¿Ha ocurrido un error en el buffer de entrada USART?
                GOTO    ESPERA          ; NO, seguir esperando
                BCF     RCSTA, CREN     ; SÍ, limpiar bit CREN del registro RCSTA
                NOP
                BSF     RCSTA, CREN     ; Reactivar bit CREN del registro RCSTA
                GOTO    ESPERA          ; Seguir esperando


; A continuación se procede a esperar la activación de alguna de las
; electroválvulas antes de capturar el promedio, para determinar a cuál
; salida corresponde ese valor capturado.


DESACT BCF      PORTD, CEL      ; Desactivar señal de habilitación del sistema

VERIF1          BTFSS   PORTA,  EV1     ; ¿La EV1 está activa?
                GOTO    VERIF2          ; NO, ir a verificar EV2
                MOVLW   0x01            ; SÍ, carga en GAS el valor que identifica
                MOVWF   GAS             ; al combustible descargado por la EV1
                GOTO    ESPIC1          ; Ir a esperar por el envío del promedio

VERIF2          BTFSS   PORTA,  EV2     ; ¿La EV2 está activa?
                GOTO    VERIF3          ; NO, ir a verificar EV3
                MOVLW   0x02            ; SÍ, carga en GAS el valor que identifica
                MOVWF   GAS             ; al combustible descargado por la EV2
                GOTO    ESPIC1          ; Ir a esperar por el envío del promedio

VERIF3          BTFSS   PORTA,   EV3    ; ¿La EV3 está activa?
                GOTO    VERIF           ; NO, ir a VERIF
                MOVLW   0x03            ; SÍ, carga en GAS el valor que identifica
                MOVWF   GAS             ; al combustible descargado por la EV3


ESPIC1          BTFSS   PORTA,  ENV     ; ¿El PIC1 indica que ya envió PROML y PROMH?
                GOTO    ESPIC1          ; NO, continuar esperando
                                        ; SÍ,


CONT            MOVF    PORTB,   w      ; Capturar el valor de PROML
                MOVWF   TOTAL1L         ; Lo guarda en memoria utilizando a TOTAL1L
```

```
                MOVF    PORTD, w        ; Lee el puerto D y limpia la lectura
                ANDLW   0x03            ; para guardar aquellos bits que corresponden
                MOVWF   TOTAL1H         ; a PROMH, utilizando a TOTAL1H.

                BSF     PORTD, REC      ; Indica al PIC1 que ya capturó PROM

ESP             BTFSC   PORTA, ENV      ; ¿La señal de envío permanece activa?
                GOTO    ESP             ; SÍ, espera que se desactive
                BCF     PORTD, REC      ; NO, desactiva la señal de recibido
```

```
;*****************************************************************************************************************
;                                       3. ELABORACIÓN DE TABLA DE DATOS
```

```
                MOVF    GAS, w          ; Traslada el tipo de combustible hacia
                MOVWF   INDF            ; la dirección que apunta el FSR.
                INCF    FSR, f          ; Incrementa el puntero del área GPR (FSR)
                CALL    AJUSTE

                MOVF    TOTAL1L, w      ; Traslada PROML hacia
                MOVWF   INDF            ; la dirección que apunta el FSR.
                INCF    FSR, f          ; Incrementa el puntero del área GPR (FSR)
                CALL    AJUSTE

                MOVF    TOTAL1H, w      ; Traslada PROMH hacia
                MOVWF   INDF            ; la dirección que apunta el FSR.
                INCF    FSR, f          ; Incrementa el puntero del área GPR (FSR)
                CALL    AJUSTE

VERIF           BTFSC   PORTA, LLAVE    ; ¿El sistema sigue habilitado por la llave?
                GOTO    VERIF1          ; SÍ, esperar por otro dato
                                        ; NO, iniciar procesamiento y envío de datos de la tabla
                CLRF    TOTAL1L         ; Limpia los registros utilizados
                CLRF    TOTAL1H         ; en la elaboración de la tabla
```

```
;*****************************************************************************************************************
;                                       4. PROCESAMIENTO DE LOS DATOS DE TABLA.
```

```
                CLRF    STATUS          ; Limpia STATUS para elegir Banco0 de Memoria.
                                        ; Además, hace que IRP=0 (Banco0/Banco1)
                MOVLW   0x28            ; 1ª dirección del área GPR que se traslada
                MOVWF   FSR             ; al Registro de Direccionamiento Indirecto

COMP1           MOVF    INDF, w         ; Carga el valor del registro apuntado por el FSR
                SUBLW   0x01            ;
                BTFSS   STATUS, Z       ; ¿El promedio corresponde a la EV1?
```

```
        GOTO    COMP2           ; NO, comprobar si es EV2
                                ; SÍ, ir a cargar los valores promedio
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte baja de un promedio en W
        ADDWF   TOTAL1L, f      ; Acumula dicho promedio al total de promedios
        BTFSC   STATUS, C       ; y ,si ocurrió un desbordamiento en dicha
        INCF    TOTAL1H, f      ; suma, incrementa la parte alta
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte alta de un promedio en W
        ADDWF   TOTAL1H, f      ; Acumula dicho promedio al total de promedios
        GOTO    SEGUIR          ; Salta a SEGUIR


COMP2   MOVF    INDF, w         ; Carga el valor del registro apuntado por el FSR
        SUBLW   0x02            ;
        BTFSS   STATUS, Z       ; ¿El promedio corresponde a la EV2?
        GOTO    COMP3           ; NO, comprobar si es EV3
                                ; SÍ, ir a cargar los valores promedio
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte baja de un promedio en W
        ADDWF   TOTAL2L, f      ; Acumula dicho promedio al total de promedios
        BTFSC   STATUS, C       ; y ,si ocurrió un desbordamiento en dicha
        INCF    TOTAL2H, f      ; suma, incrementa la parte alta
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte alta de un promedio en W
        ADDWF   TOTAL2H, f      ; Acumula dicho promedio al total de promedios
        GOTO    SEGUIR          ; Salta a SEGUIR


COMP3   MOVF    INDF, w         ; Carga el valor del registro apuntado por el FSR
        SUBLW   0x03            ;
        BTFSS   STATUS, Z       ; ¿El promedio corresponde a la EV3?
        GOTO    ENVIO           ; NO, entonces ya no hay mas datos, ir a ENVIO
                                ; SÍ, ir a cargar los valores promedio
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte baja de un promedio en W
        ADDWF   TOTAL3L, f      ; Acumula dicho promedio al total de promedios
        BTFSC   STATUS, C       ; y ,si ocurrió un desbordamiento en dicha
        INCF    TOTAL3H, f      ; suma, incrementa la parte alta
        INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        CALL    AJUSTE
        MOVF    INDF, w         ; Carga la parte alta de un promedio en W
        ADDWF   TOTAL3H, f      ; Acumula dicho promedio al total de promedios


SEGUIR  INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
        BTFSS   STATUS, IRP     ; ¿Se tiene Acceso Indirecto a los bancos 2 y 3?
        GOTO    AJUST           ; NO, continuar normalmente
        MOVF    FSR, w          ; SÍ, Compara el FSR con la dirección posterior
        SUBLW   0xF0            ; al final del área GPR del banco 3
```

```
                BTFSS   STATUS, Z       ; ¿Fin del Banco3? (¿FSR=0xF0?)
                GOTO    AJUST          ; NO, continuar normalmente
                GOTO    ENVIO          ; SÍ, ir a transmitir los totales

AJUST           CALL    AJUSTE
                GOTO    COMP1          ; Seguir con el proceso



;*****************************************************************************************************
;                                   5. ENVÍO DE DATOS HACIA EL TELEFONO


; Envío de los datos correspondientes a la EV1
ENVIO           BCF     STATUS, RP0    ; Selección de Banco0 de Memoria
                MOVLW   0x01           ; Carga en W el valor que corresponde a la EV1
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL1H, w     ; Carga en w la parte alta de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL1L, w     ; Carga en w la parte baja de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato

; Envío de los datos correspondientes a la EV2
                MOVLW   0x02           ; Carga en W el valor que corresponde a la EV1
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL2H, w     ; Carga en w la parte alta de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL2L, w     ; Carga en w la parte baja de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato

; Envío de los datos correspondientes a la EV3
                MOVLW   0x03           ; Carga en W el valor que corresponde a la EV1
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL3H, w     ; Carga en w la parte alta de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato
                MOVF    TOTAL3L, w     ; Carga en w la parte baja de la suma de promedios
                MOVWF   VAR            ; Carga en VAR el valor que será transmitido
                CALL    TRANS          ; Llama a la subrutina para enviar el dato



;*****************************************************************************************************
;                           6. VALIDACIÓN DEL TELÉFONO PARA REINICIAR EL SISTEMA


RX              BTFSS   PIR1, RCIF     ; ¿Se ha completado la recepción de un dato?
```

```
                GOTO    RX              ; NO, seguir esperando
                MOVF    RCREG, w        ; SÍ, limpiar el buffer de entrada

; Espera de 100ms para que el teléfono notifique un error en la transmisión

                BSF     T2CON, TMR2ON   ; Habilita el Timer2 e inicia la cuenta
TEMP2.1         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.1         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.2         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.2         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.3         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.3         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.4         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.4         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
                BCF     T2CON, TMR2ON   ; Desactiva el Timer2

                BTFSS   PIR1, RCIF      ; ¿Se ha recibido algún dato?
                GOTO    REINIC          ; NO, significa que no hubo error. Reiniciar el sistema
                MOVF    RCREG, w        ; SÍ, cargar el dato recibido en w para limpiar
                GOTO    ENVIO           ; el buffer de entrada. Retransmitir los datos

REINIC          CLRF    TOTAL1H         ; Se limpian las variables que almacenan los totales
                CLRF    TOTAL1L         ;
                CLRF    TOTAL2H         ;
                CLRF    TOTAL2L         ;
                CLRF    TOTAL3H         ;
                CLRF    TOTAL3L         ;
                GOTO    INICIO          ; Regresa al inicio



;*****************************************************************************************************************************
;               SUBRUTINA PARA TRANSMISIÓN SERIAL DE DATOS
;                       MEDIANTE EL MÓDULO USART.

TRANS           BCF     STATUS, RP0     ; Cambia a banco 0
                MOVF    VAR, w          ; Carga el dato a transmitir en W
                MOVWF   TXREG           ; Transmite el valor contenido en W
                BSF     STATUS, RP0     ; Cambia a banco 1
TX              BTFSS   TXSTA, TRMT     ; ¿Transmisión completa?
                GOTO    TX              ; NO, Esperar
                BCF     STATUS, RP0     ; SÍ, volver al banco 0


; Espera de 100ms para que el teléfono notifique un error en la transmisión

                BSF     T2CON, TMR2ON   ; Habilita el Timer2 e inicia la cuenta
TEMP2.5         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.5         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.6         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.6         ; NO, continuar esperando
```

```
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.7         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.7         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
TEMP2.8         BTFSS   PIR1, TMR2IF    ; ¿El TMR2 = PR2 (overflow)?
                GOTO    TEMP2.8         ; NO, continuar esperando
                BCF     PIR1, TMR2IF    ; SÍ, limpiar bandera de overflow
                BCF     T2CON, TMR2ON   ; Desactiva el Timer2

                BTFSS   PIR1, RCIF      ; ¿Se ha recibido algún dato?
                GOTO    RETORNO         ; NO, significa que no hubo error
                MOVF    RCREG, w        ; SÍ, cargar el dato recibido en w para limpiar
                GOTO    TRANS           ; el buffer de entrada y retransmitir el dato

RETORNO         RETURN


;*****************************************************************************************************************
;
;                       SUBRUTINA DE LIMPIEZA DE TODA EL AREA GPR
;                       EN LA CUAL SE ELABORARA LA TABLA DE DATOS.

LIMPIAR         BCF     STATUS, IRP     ; Se apunta indirectamente a Banco0/Banco1
                MOVLW   0x28            ; Se apunta a la primera dirección del
                MOVWF   FSR             ; área GPR en el Banco0.

BANCO0          CLRF    INDF            ; Limpiar la dirección apuntada por el FSR
                INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
                MOVF    FSR, w          ; Compara el FSR con la dirección posterior
                SUBLW   0x80            ; al final del área GPR del banco 0
                BTFSS   STATUS, Z       ; ¿Fin del Banco0? (¿FSR=0x80?)
                GOTO    BANCO0          ; NO, seguir con BANCO0
                MOVLW   0xA0            ; SÍ, apuntar hacia la primera dirección del
                MOVWF   FSR             ; área GPR del banco1.


BANCO1          CLRF    INDF            ; Limpiar la dirección apuntada por el FSR
                INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
                MOVF    FSR, w          ; Compara el FSR con la dirección posterior
                SUBLW   0xF0            ; al final del área GPR del banco 1
                BTFSS   STATUS, Z       ; ¿Fin del Banco1? (¿FSR=0xF0?)
                GOTO    BANCO1          ; NO, seguir con BANCO1
                BSF     STATUS, IRP     ; SÍ, permitir Acceso Indirecto
                                        ; hacia Banco2/Banco3.
                MOVLW   0x10            ; Apuntar hacia la 1ª dirección del área
                MOVWF   FSR             ; GPR del banco2

BANCO2 CLRF     INDF                    ; Limpiar la dirección apuntada por el FSR
                INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
                MOVF    FSR, w          ; Compara el FSR con la dirección posterior
                SUBLW   0x70            ; al final del área GPR del banco 2
                BTFSS   STATUS, Z       ; ¿Fin del Banco2? (¿FSR=0x70?)
                GOTO    BANCO2          ; NO, ir a BANCO2
                MOVLW   0x90            ; SÍ, apuntar hacia la 1ª dirección del área
                MOVWF   FSR             ; GPR del banco3

BANCO3 CLRF     INDF                    ; Limpiar la dirección apuntada por el FSR
```

```
                INCF    FSR, f          ; Incrementa el apuntador indirecto FSR
                MOVF    FSR, w          ; Compara el FSR con la dirección posterior
                SUBLW   0xF0            ; al final del área GPR del banco 3
                BTFSS   STATUS, Z       ; ¿Fin del Banco3? (¿FSR=0xF0?)
                GOTO    BANCO3          ; NO, ir a BANCO3
                                        ; SÍ, terminar subrutina

SALIR           RETURN



;*****************************************************************************************************************
;                       SUBRUTINA DE AJUSTE DE BANCO DE MEMORIA
;                       SEGUN EL VALOR DEL REGISTRO FSR.

AJUSTE          BTFSC STATUS, IRP       ; ¿Se apunta indirectamente a Banco2/Banco3?
                GOTO SALTO1             ; SÍ, ir a SALTO1
                                        ; NO, continuar

                MOVF FSR, w             ; Compara el FSR con la dirección posterior
                SUBLW 0x80              ; al final del área GPR del banco 0
                BTFSS STATUS, Z         ; ¿Fin del Banco0? (¿FSR=0x80?)
                GOTO SALTO2             ; NO, ir a SALTO2
                MOVLW 0xA0              ; SÍ, apuntar hacia la 1ª dirección del área
                MOVWF FSR               ; GPR del banco1
                GOTO SALTO3             ; Ir a SALTO3

SALTO2          MOVF FSR, w             ; Compara el FSR con la dirección posterior
                SUBLW 0xF0              ; al final del área GPR del banco 1
                BTFSS STATUS, Z         ; ¿Fin del Banco1? (¿FSR=0xF0?)
                GOTO SALTO3             ; NO, ir a SALTO3
                BSF STATUS, IRP         ; SÍ, permitir Acceso Indirecto
                                        ; hacia Banco2/Banco3.
                MOVLW 0x10              ; Apuntar hacia la 1ª dirección del área
                MOVWF FSR               ; GPR del banco2
                GOTO SALTO3             ; Ir a SALTO3


SALTO1          MOVF FSR, w             ; Compara el FSR con la dirección posterior
                SUBLW 0x70              ; al final del área GPR del banco 2
                BTFSS STATUS, Z         ; ¿Fin del Banco2? (¿FSR=0x70?)
                GOTO SALTO4             ; NO, ir a SALTO4
                MOVLW 0x90              ; SÍ, apuntar hacia la 1ª dirección del área
                MOVWF FSR               ; GPR del banco3
                GOTO SALTO3

SALTO4          MOVF FSR, w             ; Compara el FSR con la dirección posterior
                SUBLW 0xF0              ; al final del área GPR del banco 3
                BTFSS STATUS, Z         ; ¿Fin del Banco3? (¿FSR=0xF0?)
                GOTO SALTO3             ; NO, ir a SALTO3
                MOVLW 0xA5              ; SÍ, apuntar nuevamente a la dirección 0xA5
                MOVWF FSR               ; del área GPR del banco3
```

```
SALTO3          RETURN


; La capacidad del área GPR utilizada podría almacenar un máximo de información
; correspondiente a 120 minutos, ya que para cada minuto se almacenan
; 3 datos de 1 byte cada uno ( cuál electroválvula se activó, PROMH, y PROML).
; Para la elaboración de la tabla de datos se dispone de un total de 360 bytes.



                END
```

APÉNDICE J. CÓDIGO DEL MIDlet HÉRMES.

```
// Clase Hermes
package Hermes;

// Llamada de las librerias a utilizar

import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.wireless.messaging.*;
import javax.microedition.media.*;
import javax.microedition.media.control.*;
import javax.microedition.io.*;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.lang.Integer;
import java.lang.Math;
import java.util.Timer;
import java.lang.Object;
import java.util.Date;
import java.util.*;

public class Hermes extends MIDlet implements CommandListener, ItemCommandListener {

// Constructor de clases.
// Declaración de las variables a utilizar de tipo String, Integer, Date y declaraciones de las  clases.

// Datos de tipo String-----------------------------------------------------
String pass1="x";
String pass2="x";
String comparacion1="1234";         // Codigo de acceso
String comparacion2="5678";         // Codigo de usuario
String comparacion3="x";
String codestacion="x";
String telefono="77650291";         // Numero telefonico de la estacion de servicio
String borrar="";
String imei="12345678901234";     // Numero de serie propio de cada movil
String datos="x";
String combustible="RSD";           // Variable que codifica el tipo de combustible
String tipo="1";                    // tipo junto con combustible nos da el tipo
                                    // de combustible descargado
                                                // 1=Diesel,2=Regular,3=Super
String descarga="1200";
String informacion="x";                     //Variable donde se construye el sms a enviar
String smsPort="0";               // Numero de puerto sms
String user="x";

// Datos de tipo entero (integer) ------------------------------------------------
  int seleccion=999;
  int resultado1=999;
  int resultado2=999;
  int resultado3=999;
  int marca=0;
  int volumen=100;
  int vard=0;
```

```
  int num1=0;
  int num2=0;
  int outdata=0;
  int indata=0;

// Variables de entrada y salida de puerto infrarrojo ----------------------
  InputStream is;
  OutputStream os;

// Llamada de la clase sender para el envío de sms y
// Variables date para la hora del sistema ------------------------------------
  SMSSender sender;
  Date hinicio;
  Date hfin;

// Variables de para el uso de tonos ------------------------------------------
  VolumeControl vc ;
  private Object tonePlayer;
  ToneControl tc;

    public Hermes() {
        super();
       initialize();
    }

// Declaracion de clases a ser utilizadas

    private Form Principal;
    private TextBox password1;
    private TextBox password2;
    private List estacionlist;
    private Form cabanas;
    private Form ahuachapan;
    private Form chalatenango;
    private Form cuscatlan;
    private Form lalibertad;
    private Form lapaz;
    private Form launion;
    private Form morazan;
    private Form sanmiguel;
    private Form sansalvador;
    private Form sanvicente;
    private Form santana;
    private Form sonsonate;
    private Form usulutan;
    private Form infrared;
    private Form retorno;
    private Alert error1;
    private Alert error2;
    private Alert error3;
    private Alert errorMessageAlert;
    private Alert destinationAddressBox;
    private Alert sendingMessageAlert;
    private Command exitCommand1;
    private Command backCommand1;
    private Command cancelCommand1;
    private Command okCommand1;
```

```
    private Ticker ticker1;
    private Ticker ticker2;
    private Ticker ticker3;
    private Ticker ticker4;
    private Ticker ticker5;
    private ChoiceGroup choiceGroup1;
    private ChoiceGroup choiceGroup2;
    private ChoiceGroup choiceGroup3;
    private ChoiceGroup choiceGroup4;
    private ChoiceGroup choiceGroup5;
    private ChoiceGroup choiceGroup6;
    private ChoiceGroup choiceGroup7;
    private ChoiceGroup choiceGroup8;
    private ChoiceGroup choiceGroup9;
    private ChoiceGroup choiceGroup10;
    private ChoiceGroup choiceGroup11;
    private ChoiceGroup choiceGroup12;
    private ChoiceGroup choiceGroup13;
    private ChoiceGroup choiceGroup14;
    private StringItem stringItem1;
    private ImageItem imageItem1;
    private Image component1;
    private Image image1;
    private ImageItem imageItem2;
    private Image image2;
    private Command itemCommand1;


// Este metodo inicializa el UI de la aplicacion.

    private void initialize() {
        getDisplay().setCurrent(get_Principal());
    }

// Llamadas del sistema q indica cual de los botones o commands a sido invocado

    public void commandAction(Command command, Item item) {

// Llamada de command en los Choicegroup X

if (item == choiceGroup1) {
            if (command == okCommand1) {
                seleccion=choiceGroup1.getSelectedIndex();
               if(seleccion==0){
                   codestacion="anato";
                getDisplay().setCurrent(get_infrared());
               }
                if(seleccion==1){
                   codestacion="anhac";
                getDisplay().setCurrent(get_infrared());
               }
                if(seleccion==2){
                   codestacion="anchi";
                getDisplay().setCurrent(get_infrared());
               }
              }
} else if (item == choiceGroup2) {
```

```
            if (command == okCommand1) {
                seleccion=choiceGroup2.getSelectedIndex();
                if(seleccion==0){
                    codestacion="csilo";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                    codestacion="csjut";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                    codestacion="cssen";
                getDisplay().setCurrent(get_infrared());
                }
               }
} else if (item == choiceGroup3) {
            if (command == okCommand1) {
              seleccion=choiceGroup3.getSelectedIndex();
                if(seleccion==0){
                  codestacion="cgpal";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                  codestacion="cgina";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                  codestacion="cgtej";
                getDisplay().setCurrent(get_infrared());
                }
                }
        } else if (item == choiceGroup4) {
            if (command == okCommand1) {
              seleccion=choiceGroup4.getSelectedIndex();
                if(seleccion==0){
                  codestacion="ctcoj";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                  codestacion="ctsuc";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                  codestacion="ctten";
                getDisplay().setCurrent(get_infrared());
                }
             }
        } else if (item == choiceGroup5) {
            if (command == okCommand1) {
              seleccion=choiceGroup5.getSelectedIndex();
                if(seleccion==0){
                    codestacion="ldjay";
                getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                    codestacion="ldjic";
                getDisplay().setCurrent(get_infrared());
```

```
                }
                 if(seleccion==2){
                   codestacion="ldtep";
                 getDisplay().setCurrent(get_infrared());
                }
               }
        } else if (item == choiceGroup6) {
            if (command == okCommand1) {
                seleccion=choiceGroup6.getSelectedIndex();
                if(seleccion==0){
                    codestacion="paspn";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                    codestacion="pasem";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                    codestacion="pazac";
                 getDisplay().setCurrent(get_infrared());
                }
                }
         } else if (item == choiceGroup7) {
            if (command == okCommand1) {
              seleccion=choiceGroup7.getSelectedIndex();
                if(seleccion==0){
                  codestacion="uocut";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                  codestacion="uolis";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                  codestacion="uosrl";
                 getDisplay().setCurrent(get_infrared());
                }
               }
} else if (item == choiceGroup8) {
            if (command == okCommand1) {
               seleccion=choiceGroup8.getSelectedIndex();
                if(seleccion==0){
                   codestacion="mzcac";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==1){
                   codestacion="mzgua";
                 getDisplay().setCurrent(get_infrared());
                }
                 if(seleccion==2){
                   codestacion="mztor";
                 getDisplay().setCurrent(get_infrared());
                }
               }
} else if (item == choiceGroup9)
            if (command == okCommand1) {
               seleccion=choiceGroup9.getSelectedIndex();
```

```
        if(seleccion==0){
          codestacion="sgcap";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==1){
          codestacion="sgcba";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==2){
          codestacion="sgmon";
         getDisplay().setCurrent(get_infrared());
        }
    }
} else if (item == choiceGroup10) {
    if (command == okCommand1) {
       seleccion=choiceGroup10.getSelectedIndex();
        if(seleccion==0){
          codestacion="sdpan";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==1){
          codestacion="sdsoy";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==2){
          codestacion="sdton";
         getDisplay().setCurrent(get_infrared());
        }
     }
} else if (item == choiceGroup11) {
    if (command == okCommand1) {
       seleccion=choiceGroup11.getSelectedIndex();
        if(seleccion==0){
          codestacion="stsid";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==1){
          codestacion="stsse";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==2){
          codestacion="stsci";
         getDisplay().setCurrent(get_infrared());
        }
      }
} else if (item == choiceGroup12) {
    if (command == okCommand1) {
       seleccion=choiceGroup12.getSelectedIndex();
        if(seleccion==0){
          codestacion="sncoa";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==1){
          codestacion="snmet";
         getDisplay().setCurrent(get_infrared());
        }
         if(seleccion==2){
```

```
                codestacion="snsrg";
               getDisplay().setCurrent(get_infrared());
              }
              }
       } else if (item == choiceGroup13) {
          if (command == okCommand1) {
             seleccion=choiceGroup13.getSelectedIndex();
              if(seleccion==0){
                codestacion="saaca";
               getDisplay().setCurrent(get_infrared());
              }
               if(seleccion==1){
                codestacion="saiza";
               getDisplay().setCurrent(get_infrared());
              }
               if(seleccion==2){
                codestacion="sasal";
               getDisplay().setCurrent(get_infrared());
              }
             }
} else if (item == choiceGroup14) {
               if (command == okCommand1) {              seleccion=choiceGroup14.getSelectedIndex();
              if(seleccion==0){
                codestacion="ulber";
               getDisplay().setCurrent(get_infrared());
              }
               if(seleccion==1){
                codestacion="ulesp";
               getDisplay().setCurrent(get_infrared());
              }
               if(seleccion==2){
                codestacion="ulele";
               getDisplay().setCurrent(get_infrared());
              }
              }
       }
}

// Llamadas del sistema que indica cual de los botones o commands a sido invocado

    public void commandAction(Command command, Displayable displayable){

 // Llamada de command en los TextBox y de Form_List_elements

// OkCommand de password1

if (displayable == password1) {
          if (command == okCommand1) {

//Se obtiene el codigo de acceso que ha tecleado el usuario.
               pass1=password1.getString();

//Se compara el código de acceso introducido por el usuario con previamente determinado.
               resultado1= pass1.compareTo(comparacion1);

// Si el codigo es correcto: se guarda la hora, se convierte el codigo de usuario de string a integer,  se limpia la pantalla y se pasa a las siguiente password2.
```

```
        if (resultado1==0){
            hinicio=new Date();
            num1= Integer.valueOf(pass1).intValue();
            password1.setString("");
            getDisplay().setCurrent(get_password2());
           }

// Si el codigo es incorrecto: se limpia la pantalla se muestra un mensaje de error y se vuelve a presentar la misma pantalla
para que ingrese el codigo nuevamente.
            if (resultado1>0){
            password1.setString("");
            getDisplay().setCurrent(get_error1(), get_password1());
            }
            if (resultado1<0){
            password1.setString("");
            getDisplay().setCurrent(get_error3(), get_password1());
            password1.setString("");
            }

// OkCommand de password2

    } else if (displayable == password2) {
          if (command == okCommand1) {

//Se obtiene el codigo de usuario que ha tecleado el usuario.
            pass2=password2.getString();

//Se compara el código de usuario introducido por el usuario con previamente determinado.
              resultado2= pass2.compareTo(comparacion2);

// Si el codigo es correcto: se guarda el codigo de usuario, se convierte el codigo de usuario de string a integer,  se limpia la
pantalla y se pasa a la lista de estaciones.

             if (resultado1==0){
             user=pass2;
             num2= Integer.valueOf(pass2).intValue();
            password2.setString("");
               getDisplay().setCurrent(get_estacionlist());
             }

// Si el codigo es incorrecto: se limpia la pantalla se muestra un mensaje de error y se vuelve a presentar la misma pantalla
para que ingrese el codigo nuevamente.
            if (resultado2>0){
            password2.setString("");
            getDisplay().setCurrent(get_error2(), get_password2());
            password1.setString("");
            }
            if (resultado2<0){
            password2.setString("");
            getDisplay().setCurrent(get_error3(), get_password2());
            password1.setString("");
            }

// Si se presiona el boton de exit para password 1 y 2 retorna a principal
          } else if (command == exitCommand1) {
            getDisplay().setCurrent(get_Principal());
          }
```

```
            } else if (command == exitCommand1) {
                getDisplay().setCurrent(get_Principal());
            }

// Commands de Principal

        } else if (displayable == Principal) {

// Si se presiona ok mostrara password1 y al presionar exit cierra la aplicacion
            if (command == okCommand1) {
                getDisplay().setCurrent(get_password1());
            } else if (command == exitCommand1) {
                exitMIDlet();
            }

// OkCommand de Form_List_elements; al seleccionar una opcion de la lista este mostrara el Form de la lista.
        } else if (displayable == estacionlist) {
            if (command == okCommand1) {
                switch (get_estacionlist().getSelectedIndex()) {
                    case 0:
                        getDisplay().setCurrent(get_ahuachapan());
                        break;
                    case 1:
                        getDisplay().setCurrent(get_cabanas());
                        break;
                    case 2:
                        getDisplay().setCurrent(get_chalatenango());
                        break;
                    case 3:
                        getDisplay().setCurrent(get_cuscatlan());
                        break;
                    case 4:
                            getDisplay().setCurrent(get_lalibertad());
                        break;
                    case 5:
                        getDisplay().setCurrent(get_lapaz());
                        break;
                    case 6:
                        getDisplay().setCurrent(get_launion());
                        break;
                    case 7:
                        getDisplay().setCurrent(get_morazan());
                        break;
                    case 8:
                        getDisplay().setCurrent(get_sanmiguel());
                        break;
                    case 9:
                        getDisplay().setCurrent(get_sansalvador());
                        break;
                    case 10:
                        getDisplay().setCurrent(get_sanvicente());
                        break;
                    case 11:
                        getDisplay().setCurrent(get_santana());
                        break;
                    case 12:
                        getDisplay().setCurrent(get_sonsonate());
```

```
                    break;
                case 13:
                    getDisplay().setCurrent(get_usulutan());
                    break;
            }

// exitCommand de Form_List_elements; al presionar este este hace que retorne a principal.
        } else if (command == exitCommand1) {
            getDisplay().setCurrent(get_Principal());
        }
// backCommand de Form_List_elements; hace regresar a la anterior Form_List_elements;
    } else if (displayable == ahuachapan) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == sanmiguel) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == sansalvador) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == launion) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == sanvicente) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == morazan) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == chalatenango) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == cuscatlan) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == cabanas) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == sonsonate) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == santana) {
        if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
        }
    } else if (displayable == lalibertad) {
        if (command == backCommand1) {
```

```
            getDisplay().setCurrent(get_estacionlist());
         }
      } else if (displayable == lapaz) {
         if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
         }
      } else if (displayable == usulutan) {
         if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());
         }

// Commands de infrared
      } else if (displayable == infrared) {
         if (command == backCommand1) {
            getDisplay().setCurrent(get_estacionlist());

// OkCommand de infrared
         } else if (command == okCommand1) {

// Creacion de un tono y una secuencia de tonos.
   byte tempo = 30;                                    // coloca tempo a 120 bpm
   byte d = 8;                                         // ocho notas
   byte C4 = ToneControl.C4;;
   byte D4 = (byte)(C4 + 2);
   byte E4 = (byte)(C4 + 4);
   byte G4 = (byte)(C4 + 7);
   byte rest = ToneControl.SILENCE;                    // pausa
   byte[] mySequence = {
      ToneControl.VERSION, 1,                          // version 1
      ToneControl.TEMPO, tempo,                        // coloca tempo
      ToneControl.BLOCK_START, 0,                      // inicia la seccion “A”
      ToneControl.SET_VOLUME,100,                      // coloca el volumen en maximo
      E4,d, D4,d, C4,d, E4,d,                          // Contenido de la Seccion “A”
      E4,d, E4,d, E4,d, rest,d,
      ToneControl.BLOCK_END, 0,                        // fin de la seccion “A”
      ToneControl.PLAY_BLOCK, 0,     // play seccion "A"
      D4,d, D4,d, D4,d, rest,d,                        // play seccion "B"
      E4,d, G4,d, G4,d, rest,d,
      ToneControl.PLAY_BLOCK, 0,                       // repetir seccion "A"
      D4,d, D4,d, E4,d, D4,d, C4,d                     // play seccion "C"
   };

// Se llama la secuencia de tono y se toca
   try{
      Player p = Manager.createPlayer(Manager.TONE_DEVICE_LOCATOR);
      p.realize();
      ToneControl c = (ToneControl)p.getControl("ToneControl");
      c.setSequence(mySequence);
      p.start();
   } catch (IOException ioe) { } catch (MediaException me) { }

// Se obtiene la hora actual, se divide num1/100 y num2/100 y se suman los resultados.
            hfin=new Date();
            num1=num1/100;
            num2=num2/100;
            vard=num1+num2;
```

```
// Configuracion el puerto infrarrojo, configura la variable de salidda y de entrada, abre el puerto
// infrarrojo y envia la vard.
 try {
        try {
        StreamConnection sc=(StreamConnection)Connector.open("nokiacomm:IR0");
        is = sc.openInputStream();
        os = sc.openOutputStream();
        } catch (ConnectionNotFoundException cnfe) {
          cnfe.printStackTrace();
        }
        os.write(vard);
    } catch (IOException ex) {}

// Construccion de sms de la siguiente forma= combustible,tipo,imei,codigo de la estacion
// descarga realizada, codigo de usuario, hora de inicio de descarga y hora de finalizacion de
// descarga
// Luego se activa la clase sender para activar el envio de sms
             informacion =combustible+tipo+imei+codestacion+descarga+user+hinicio+hfin;
             sender = new SMSSender(smsPort, telefono,informacion, retorno, infrared);
             promptAndSend();

           }
       } else if (displayable == retorno) {
           if (command == itemCommand1)
              exitMIDlet
           }
       }
}


    public Display getDisplay() {
        return Display.getDisplay(this);
    }

    public void exitMIDlet() {
        getDisplay().setCurrent(null);
        destroyApp(true);
        notifyDestroyed();
    }

// Construccion de el objeto Form principal.

    public Form get_Principal() {
        if (Principal == null) {
            marca=1;
            Principal = new Form("Informacion", new Item[] {get_stringItem1()});
            Principal.addCommand(get_exitCommand1());
            Principal.addCommand(get_okCommand1());
            Principal.setCommandListener(this);
            Principal.setTicker(get_ticker4());
        }
        return Principal;
    }

// Construccion de el objeto TextBoxt password1.
    public TextBox get_password1() {
        if (password1 == null) {
```

```
            marca=2;
            password1 = new TextBox("Ingrese el codigo del acceso:", "", 4, TextField.NUMERIC);
            password1.addCommand(get_exitCommand1());
            password1.addCommand(get_okCommand1());
            password1.setCommandListener(this);
            password1.setTicker(get_ticker2());
        }
        return password1;
    }


    // Construccion de el objeto TextBox passwor2.
    public TextBox get_password2() {
        if (password2 == null) {
            marca=3;
            password2 = new TextBox("Ingrese el codigo del usuario:", "", 4, TextField.NUMERIC);
            password2.addCommand(get_exitCommand1());
            password2.addCommand(get_okCommand1());
            password2.setCommandListener(this);
            password2.setTicker(get_ticker3());
        }
        return password2;
    }

    // Construccion de el objeto List estacionlist.
    public List get_estacionlist() {
        if (estacionlist == null) {
            marca=4;
            estacionlist = new List(null, Choice.IMPLICIT, new String[] {
                "Ahuachapan",
                        "Cabanas",
                        "Chalatenango",
                        "Cuscatlan",
                        "La Libertad",
                        "La Paz",
                        "La Union",
                        "Morazan",
                        "San Miguel",
                        "San Salvador",
                        "San Vicente",
                        "Santa Ana",
                        "Sonsonate",
                        "Usulutan"
            }, new Image[] {
                null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
                        null,
```

```
                null
        });
        estacionlist.addCommand(get_exitCommand1());
        estacionlist.setCommandListener(this);
        estacionlist.setTicker(get_ticker1());
        estacionlist.setSelectedFlags(new boolean[] {
           false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false,
                false
        });
        estacionlist.setSelectCommand(get_okCommand1());
    }
    return estacionlist;
}

// Construccion de el objeto Form cabanas.
public Form get_cabanas() {
    if (cabanas == null) {
        marca=6;
        cabanas = new Form(null, new Item[] {get_choiceGroup2()});
        cabanas.addCommand(get_backCommand1());
        cabanas.setCommandListener(this);
        cabanas.setTicker(get_ticker5());
    }
    return cabanas;
}

// Construccion de el objeto Form ahuachapan.
public Form get_ahuachapan() {
    if (ahuachapan == null) {
        marca=5;
        ahuachapan = new Form(null, new Item[] {get_choiceGroup1()});
        ahuachapan.addCommand(get_backCommand1());
        ahuachapan.setCommandListener(this);
        ahuachapan.setTicker(get_ticker5());
    }
    return ahuachapan;
}

// Construccion de el objeto Form chalatenango.
public Form get_chalatenango() {
    if (chalatenango == null) {
        marca=7;
        chalatenango = new Form(null, new Item[] {get_choiceGroup3()});
        chalatenango.addCommand(get_backCommand1());
        chalatenango.setCommandListener(this);
```

```
            chalatenango.setTicker(get_ticker5());
        }
        return chalatenango;
    }

    // Construccion de el objeto Form cuscatlan
    public Form get_cuscatlan() {
        if (cuscatlan == null) {
            marca=8;
            cuscatlan = new Form(null, new Item[] {get_choiceGroup4()});
            cuscatlan.addCommand(get_backCommand1());
            cuscatlan.setCommandListener(this);
            cuscatlan.setTicker(get_ticker5());
        }
        return cuscatlan;
    }

    // Construccion de el objeto Form lalibertad
    public Form get_lalibertad() {
        if (lalibertad == null) {
            marca=9;
            lalibertad = new Form(null, new Item[] {get_choiceGroup5()});
            lalibertad.addCommand(get_backCommand1());
            lalibertad.setCommandListener(this);
            lalibertad.setTicker(get_ticker5());
        }
        return lalibertad;
    }

    // Construccion de el objeto Form lapa
    public Form get_lapaz() {
        if (lapaz == null) {
            marca=10;
            lapaz = new Form(null, new Item[] {get_choiceGroup6()});
            lapaz.addCommand(get_backCommand1());
            lapaz.setCommandListener(this);
            lapaz.setTicker(get_ticker5());
        }
        return lapaz;
    }

    // Construccion de el objeto Form launion
    public Form get_launion() {
        if (launion == null) {
            marca=11;
            launion = new Form(null, new Item[] {get_choiceGroup7()});
            launion.addCommand(get_backCommand1());
            launion.setCommandListener(this);
            launion.setTicker(get_ticker5());
        }
        return launion;
    }

// Construccion de el objeto Form morazan
    public Form get_morazan() {
        if (morazan == null) {
            marca=12;
```

```
            morazan = new Form(null, new Item[] {get_choiceGroup8()});
            morazan.addCommand(get_backCommand1());
            morazan.setCommandListener(this);
            morazan.setTicker(get_ticker5());
        }
        return morazan;
    }

    // Construccion de el objeto Form sanmiguel.
    public Form get_sanmiguel() {
        if (sanmiguel == null) {
            marca=13;
            sanmiguel = new Form(null, new Item[] {get_choiceGroup9()});
            sanmiguel.addCommand(get_backCommand1());
            sanmiguel.setCommandListener(this);
            sanmiguel.setTicker(get_ticker5());
        }
        return sanmiguel;
    }

    // Construccion de el objeto Form sansalvador
    public Form get_sansalvador() {
        if (sansalvador == null) {
            marca=14;
            sansalvador = new Form(null, new Item[] {get_choiceGroup10()});
            sansalvador.addCommand(get_backCommand1());
            sansalvador.setCommandListener(this);
            sansalvador.setTicker(get_ticker5());
        }
        return sansalvador;
    }

// Construccion de el objeto Form sanvicente.
    public Form get_sanvicente() {
        if (sanvicente == null)
            marca=15;
            sanvicente = new Form(null, new Item[] {get_choiceGroup11
            sanvicente.addCommand(get_backCommand1());
            sanvicente.setCommandListener(this);
            sanvicente.setTicker(get_ticker5
        }
        return sanvicente;
    }

    // Construccion de el objeto Form santana
    public Form get_santana() {
        if (santana == null) {
            marca=16;
            santana = new Form(null, new Item[] {get_choiceGroup12()});
            santana.addCommand(get_backCommand1());
            santana.setCommandListener(this);
            santana.setTicker(get_ticker5());
        }
        return santana;
    }

   // Construccion de el objeto Form sonsonate
```

```
    public Form get_sonsonate() {
        if (sonsonate == null) {
            marca=17;
            sonsonate = new Form(null, new Item[] {get_choiceGroup13()});
            sonsonate.addCommand(get_backCommand1());
            sonsonate.setCommandListener(this);
            sonsonate.setTicker(get_ticker5());
        }
        return sonsonate;
    }

    // Construccion de el objeto Form usulutan
    public Form get_usulutan() {
        if (usulutan == null) {
            marca=18;
            usulutan = new Form(null, new Item[] {get_choiceGroup14()});
            usulutan.addCommand(get_backCommand1());
            usulutan.setCommandListener(this);
            usulutan.setTicker(get_ticker5());
        }
        return usulutan;
    }

// Construccion de el objeto Form infrared
    public Form get_infrared() {
        if (infrared == null) {
            marca=19;
            infrared = new Form("Comunicacion", new Item[] {get_imageItem1()});
            infrared.addCommand(get_okCommand1());
            infrared.addCommand(get_backCommand1());
            infrared.setCommandListener(this);
        }
        return infrared;
    }

// Construccion de el objeto From retorno.
    public Form get_retorno() {
        if (retorno == null) {
          marca=20;
          retorno = new Form("Proceso", new Item[] {get_imageItem2()});
          retorno.addCommand(get_itemCommand1());
          retorno.setCommandListener(this);
        }
        return retorno;
    }

// Construccion de el objeto Alert error1
    public Alert get_error1() {
        if (error1 == null) {
            error1 = new Alert("Error.", "Codigo de acceso incorrecto intentelo nuevamente.", null, AlertType.ERROR);
            error1.setTimeout(3000);
        }
        return error1;
    }

    // Construccion de el objeto Aliert error2
    public Alert get_error2() {
```

```
        if (error2 == null) {
            error2 = new Alert("Error", "Codigo de usuario incorrecto ingrselo nuevamente.", null, AlertType.ERROR);
            error2.setTimeout(3000);
        }
        return error2;
    }

    // Construccion de el objeto Alert error3
    public Alert get_error3() {
        if (error3 == null) {
            error3 = new Alert("Error", "No ha ingresado ningun dato.", null, AlertType.ERROR);
            error3.setTimeout(3000);
        }
        return error3;
    }

    // Construccion de el objeto Alert error Message.
    public Alert get_errorMessageAlert() {
        if (errorMessageAlert == null) {
            errorMessageAlert = new Alert(null, "<Enter Text>", null, AlertType.ERROR);
            errorMessageAlert.setTimeout(3000);
        }
        return errorMessageAlert;
    }

// Construccion del objeto Alert destinationAddress
    public Alert get_destinationAddressBox() {
        if (destinationAddressBox == null)
            destinationAddressBox = new Alert(null, "<Enter Text>", null, AlertType.INFO);
            destinationAddressBox.setTimeout(3000);
        }
        return destinationAddressBox;
    }

    // Construccion del objeto Alert sendingMessage
    public Alert get_sendingMessageAlert() {
        if (sendingMessageAlert == null) {
            sendingMessageAlert = new Alert("Send", "Enviando mensaje espere por favor.", null, AlertType.INFO);
            sendingMessageAlert.setTimeout(3000);
        }
        return sendingMessageAlert;
    }

    public Command get_exitCommand1() {
        if (exitCommand1 == null) {
            exitCommand1 = new Command("Exit", Command.EXIT, 1);
        }
        return exitCommand1;
    }

    public Command get_backCommand1() {
        if (backCommand1 == null) {
            backCommand1 = new Command("Back", Command.BACK, 1);
        }
        return backCommand1;
    }
```

```
    public Command get_cancelCommand1() {
        if (cancelCommand1 == null) {
            cancelCommand1 = new Command("Cancel", Command.CANCEL, 1);
        }
        return cancelCommand1;
    }

    public Command get_okCommand1() {
        if (okCommand1 == null) {
            okCommand1 = new Command("Ok", Command.OK, 1);
        }
        return okCommand1;
    }

// Construccion de los objetos Tickers
    public Ticker get_ticker1() {
        if (ticker1 == null) {
            ticker1 = new Ticker("Seleccione el departamento de la estacion de servicio:");
        }
        return ticker1;
    }
    public Ticker get_ticker2() {
        if (ticker2 == null) {
            ticker2 = new Ticker("Ingrese el codigo del acceso:");
        }
        return ticker2;
    }
    public Ticker get_ticker3() {
        if (ticker3 == null) {
            ticker3 = new Ticker("Ingrese el codigo del usuario:");
        }
        return ticker3;
    }
    public Ticker get_ticker4() {
        if (ticker4 == null) {
            ticker4 = new Ticker("Pasos de inicio de programa:\n");
        }
        return ticker4;
    }
    public Ticker get_ticker5() {
        if (ticker5 == null) {
            ticker5 = new Ticker("Seleccione la estacion de servicio:");
        }
        return ticker5;
    }

// Construccion de los objeto ChoiceGroup
    public ChoiceGroup get_choiceGroup1() {
        if (choiceGroup1 == null) {
            choiceGroup1 = new ChoiceGroup("Ahuachapan", Choice.EXCLUSIVE, new String[] {
                "Ataco",
                        "La Hachadura",
                        "Las Chinamas"
            }, new Image[] {
                null,
                        null,
                        null
```

```
        });
        choiceGroup1.addCommand(get_okCommand1());
        choiceGroup1.setItemCommandListener(this);
        choiceGroup1.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup1;
}

public ChoiceGroup get_choiceGroup2() {
    if (choiceGroup2 == null) {
        choiceGroup2 = new ChoiceGroup("Cabanas", Choice.EXCLUSIVE, new String[] {
            "Ilobasco",
                "Jutiapa",
                "Sensuntepeque"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup2.addCommand(get_okCommand1());
        choiceGroup2.setItemCommandListener(this);
        choiceGroup2.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup2;
}

public ChoiceGroup get_choiceGroup3() {
    if (choiceGroup3 == null) {
        choiceGroup3 = new ChoiceGroup("Chalatenango\n", Choice.EXCLUSIVE, new String[] {
            "La Palma",
                "San Ignacio",
                "Tejutla"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup3.addCommand(get_okCommand1());
        choiceGroup3.setItemCommandListener(this);
        choiceGroup3.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup3;
}

    public ChoiceGroup get_choiceGroup4() {
```

```
    if (choiceGroup4 == null) {
        choiceGroup4 = new ChoiceGroup("Cuscatlan", Choice.EXCLUSIVE, new String[] {
            "Cojutepeque",
                    "Suchitoto",
                    "Tenancingo"
        }, new Image[] {
            null,
                    null,
                    null
        });
        choiceGroup4.addCommand(get_okCommand1());
        choiceGroup4.setItemCommandListener(this);
        choiceGroup4.setSelectedFlags(new boolean[] {
            false,
                    false,
                    false
        });
    }
    return choiceGroup4;
}


public ChoiceGroup get_choiceGroup5() {
    if (choiceGroup5 == null) {
        choiceGroup5 = new ChoiceGroup("La Libertad", Choice.EXCLUSIVE, new String[] {
            "Jayaque",
                    "Jicalapa",
                    "Tepecoyo"
        }, new Image[] {
            null,
                    null,
                    null
        });
        choiceGroup5.addCommand(get_okCommand1());
        choiceGroup5.setItemCommandListener(this);
        choiceGroup5.setSelectedFlags(new boolean[] {
            false,
                    false,
                    false
        });
    }
    return choiceGroup5;
}

public ChoiceGroup get_choiceGroup6() {
    if (choiceGroup6 == null) {
      choiceGroup6 = new ChoiceGroup("La Paz", Choice.EXCLUSIVE, new String[] {
            "San Pedro Nonualco",
                    "San Emigdio",
                    "Zacatecoluca"
        }, new Image[] {
            null,
                    null,
                    null
        });
        choiceGroup6.addCommand(get_okCommand1());
        choiceGroup6.setItemCommandListener(this);
```

```
        choiceGroup6.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup6;
}

public ChoiceGroup get_choiceGroup7() {
    if (choiceGroup7 == null) {
        choiceGroup7 = new ChoiceGroup("La Union", Choice.EXCLUSIVE, new String[] {
            "Cutuco",
                "Lislique",
                "Santa Rosa de Lima"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup7.addCommand(get_okCommand1());
        choiceGroup7.setItemCommandListener(this);
        choiceGroup7.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup7;
}

public ChoiceGroup get_choiceGroup8() {
    if (choiceGroup8 == null) {
        choiceGroup8 = new ChoiceGroup("Morazan", Choice.EXCLUSIVE, new String[] {
            "Cacaopera",
                "Gualalococti",
                "Torola"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup8.addCommand(get_okCommand1());
        choiceGroup8.setItemCommandListener(this);
        choiceGroup8.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup8;
}

public ChoiceGroup get_choiceGroup9() {
    if (choiceGroup9 == null) {
        choiceGroup9 = new ChoiceGroup("San Miguel", Choice.EXCLUSIVE, new String[] {
            "Capeltique",
```

```
                "Ciudad Barrios",
                "Moncahua"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup9.addCommand(get_okCommand1());
        choiceGroup9.setItemCommandListener(this);
        choiceGroup9.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup9;
}

public ChoiceGroup get_choiceGroup10() {
    if (choiceGroup10 == null) {
        choiceGroup10 = new ChoiceGroup("San Salvador", Choice.EXCLUSIVE, new String[] {
            "Panchimalco",
                "Soyapango",
                "Tonacatepeque"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup10.addCommand(get_okCommand1());
        choiceGroup10.setItemCommandListener(this);
        choiceGroup10.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup10;
}

public ChoiceGroup get_choiceGroup11() {
    if (choiceGroup11 == null) {
        choiceGroup11 = new ChoiceGroup("San Vicente", Choice.EXCLUSIVE, new String[] {
            "San Idelfonso",
                "San Sebastian",
                "Sn. Cayetano Ixtepeque"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup11.addCommand(get_okCommand1());
        choiceGroup11.setItemCommandListener(this);
        choiceGroup11.setSelectedFlags(new boolean[] {
            false,
                false,
                false
```

```
        });
    }
    return choiceGroup11;
}

    public ChoiceGroup get_choiceGroup12() {
    if (choiceGroup12 == null) {
        choiceGroup12 = new ChoiceGroup("Santa Ana", Choice.EXCLUSIVE, new String[] {
            "Coatepeque",
                "Metapan",
                "Santa Rosa de Guachipilin"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup12.addCommand(get_okCommand1());
        choiceGroup12.setItemCommandListener(this);
        choiceGroup12.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup12;
}

public ChoiceGroup get_choiceGroup13() {
    if (choiceGroup13 == null) {
        choiceGroup13 = new ChoiceGroup("Sonsonate", Choice.EXCLUSIVE, new String[] {
            "Acajutla",
                "Izalco",
                "Salcoatitan"
        }, new Image[] {
            null,
                null,
                null
        });
        choiceGroup13.addCommand(get_okCommand1());
        choiceGroup13.setItemCommandListener(this);
        choiceGroup13.setSelectedFlags(new boolean[] {
            false,
                false,
                false
        });
    }
    return choiceGroup13;
}

public ChoiceGroup get_choiceGroup14() {
    if (choiceGroup14 == null) {
        choiceGroup14 = new ChoiceGroup("Usulutan", Choice.EXCLUSIVE, new String[] {
            "Berlin",
                "El Espino",
                "Santa Elena"
        }, new Image[] {
            null,
```

```
                    null,
                    null
            });
            choiceGroup14.addCommand(get_okCommand1());
            choiceGroup14.setItemCommandListener(this);
            choiceGroup14.setSelectedFlags(new boolean[] {
                false,
                    false,
                    false
            });
        }
        return choiceGroup14;
    }

// Construccion del objeto StringItem
    public StringItem get_stringItem1() {
        if (stringItem1 == null) {
       stringItem1 = new StringItem("\n", "1. Ingrese el codigo de acceso del programa.\n2. Ingrese el codigo de usuario.\n3. Seleccione de una lista el departamento de la estacion de servicio.\n4. Seleccione la estacion de servicio donde se realizara la descarga.\n5. Luego el programa le indicara cuando enviara el mensaje de texto.\n6. El programa se cerrara.");
        }
        return stringItem1;
    }

// Construccion del objeto StringItem
 public ImageItem get_imageItem1() {
        if (imageItem1 == null) {
            imageItem1 = new ImageItem("", get_component1(), 0x6b33, null);
        }
        return imageItem1;
    }

// Construccion de los objetos Image
    public Image get_component1() {
        if (component1 == null) {
            try {
                component1 = Image.createImage("/Items/irda.PNG");
            } catch (java.io.IOException exception) {
            }
        }
        return component1;
    }
    public Image get_image1() {
        if (image1 == null) {
            try {//GEN-BEGIN:MVDGetInit186
                image1 = Image.createImage("<No Image>");
            } catch (java.io.IOException exception) {
            }
        }
        return image1;
    }
    public Image get_image2() {
        if (image2 == null) {
            try {
                image2 = Image.createImage("/Items/retorno.PNG");
            } catch (java.io.IOException exception) {
            }
```

```
        }
        return image2;
    }

// Construccion de el objeto ImageItem
    public ImageItem get_imageItem2() {
        if (imageItem2 == null) {
            imageItem2 = new ImageItem("El proceso ha finalizado.", get_image2(), 0x7f33, null);
        }
        return imageItem2;
    }

    public Command get_itemCommand1() {
        if (itemCommand1 == null) {
            itemCommand1 = new Command("Quit", Command.EXIT, 1);
        }
        return itemCommand1;
    }


    public void startApp() {
    }

// Codigo a ser ejecutado cuando el movil entre en estado de pausa, lo que se hace aca es mantenerse en la Form que esta
antes de la pausa.

     public void pauseApp() {
  if (marca==1){
  getDisplay().setCurrent(get_Principal());
  }
  if (marca==2){
  getDisplay().setCurrent(get_password1());
  }
  if (marca==3){
  getDisplay().setCurrent(get_password2());
  }
  if (marca==4){
  getDisplay().setCurrent(get_estacionlist());
  }
   if (marca==5){
  getDisplay().setCurrent(get_ahuachapan());
  }
   if (marca==6){
  getDisplay().setCurrent(get_cabanas());
  }
   if (marca==7){
  getDisplay().setCurrent(get_chalatenango());
  }
   if (marca==8){
  getDisplay().setCurrent(get_cuscatlan());
  }
   if (marca==9){
  getDisplay().setCurrent(get_lalibertad());
  }
   if (marca==10){
  getDisplay().setCurrent(get_lapaz());
  }
```

```
 if (marca==11){
getDisplay().setCurrent(get_launion());
}
 if (marca==12){
getDisplay().setCurrent(get_morazan());
}
 if (marca==13){
getDisplay().setCurrent(get_sanmiguel());
}
 if (marca==14){
getDisplay().setCurrent(get_sansalvador());
}
 if (marca==15){
getDisplay().setCurrent(get_sanvicente());
}
 if (marca==16){
getDisplay().setCurrent(get_santana());
}
 if (marca==17){
getDisplay().setCurrent(get_sonsonate());
}
 if (marca==18){
getDisplay().setCurrent(get_usulutan());
}
 if (marca==19){
getDisplay().setCurrent(get_infrared());
}
if (marca==20){
getDisplay().setCurrent(get_retorno());
}
  }

  public void destroyApp(boolean unconditional) {
  }

// Este verifica si el numero de telefono es valido y llama la clase sender para enviar la información.
   private void promptAndSend() {
       if (!Hermes.isValidPhoneNumber(telefono)) {
          errorMessageAlert.setString("Invalid phone number");
          return;
       }
       String statusMessage = "Sending message to " + telefono + "...";
       getDisplay().setCurrent(get_sendingMessageAlert(),get_retorno());
       sendingMessageAlert.setString(statusMessage);
       sender.promptAndSend("sms://" + telefono);
   }

   private static boolean isValidPhoneNumber(String number) {
       char[] chars = number.toCharArray();
       if (chars.length == 0) {
          return false;
       }
       int startPos = 0;
       // initial '+' is OK
       if (chars[0] == '+') {
          startPos = 1;
       }
```

```
        for (int i = startPos; i < chars.length; ++i) {
            if (!Character.isDigit(chars[i])) {
                return false;
            }
        }
        return true;
    }
}


//Clase SMSSender

package Hermes;

// Llamada de las librerias a utilizar
import javax.microedition.io.*;
import javax.microedition.lcdui.*;
import javax.wireless.messaging.*;
import java.io.IOException;

public class SMSSender  implements  Runnable {

// Declaracion de variables
    Display display;
    String envio="x";
    String smsPort;
    String informacion;
    String telefono="x";
    TextBox messageBox;
    Displayable backScreen;
    Displayable sendingScreen;
    Displayable retorno;
    Displayable infrared;

public SMSSender(String smsPort,String telefono,String informacion,Displayable retorno,Displayable infrared) {

        this.smsPort =null;
        this.telefono = null;
        this.informacion = informacion;
        this.display = display;
        this.backScreen = backScreen;
        this.sendingScreen = sendingScreen;
        this.retorno = retorno;
        this.infrared = infrared;
    }

    public void promptAndSend(String telefono)    {
         this.telefono = telefono;
         this.informacion = informacion;
         this.smsPort = smsPort;
         new Thread(this).start();
    }

// Configuracion de la conexion sms, apertura y envio de la informacion.
       public void run() {
        smsPort="0";
        String address = telefono + ":" + smsPort;
```

```
        MessageConnection smsconn = null;
        try {
             smsconn = (MessageConnection)Connector.open(address);
            TextMessage txtmessage = (TextMessage)smsconn.newMessage(MessageConnection.TEXT_MESSAGE);
            txtmessage.setAddress(address);
            txtmessage.setPayloadText(informacion);
            smsconn.send(txtmessage);
            envio="1";
        } catch (Throwable t) {
System.out.println("Se cancelo el envio del mensaje de texto ");
t.printStackTrace();
        }

        if (smsconn != null) {
            try {
                smsconn.close();
            } catch (IOException ioe) {
System.out.println("Closing connection caught: ");
ioe.printStackTrace();
            }
        }
    }
}
```

APÉNDICE K. CÓDIGO DEL SOFTWARE ADMINISTRATIVO.

```
' Declaracion de las variables

Option Explicit
' Controles a utilizar para el proceso de los sms
Public objGsmIn As ASmsCtrl.GsmIn
Public objConstants As ASmsCtrl.Constants
Public objGsmOut As ASmsCtrl.GsmOut

' Declaracion del objeto a utilizar
Public appxls1 As Excel.Application
Public data1 As Excel.Workbook
Public appxls2 As Excel.Application
Public appxls3 As Excel.Application
Public data2 As Excel.Workbook
Public data3 As Excel.Workbook
Public appxls4 As Excel.Application
Public data4 As Excel.Workbook

' Variables utilizadas para el proyecto

Dim darth As Integer                    'Lleva el numero de veces que se guarda la tabla del dia
Dim mensaje As String                   'Contiene el mensaje complete leido del movil
Dim imei As String                      'Numero de serie del movil
Dim tipo As String                      'tipo de combustible
Dim fecha As String                     'fecha del dia que se hace la tabla
Dim hora As String                      'lleva el registro de la hora
Dim descarga As String                  'cantidad de combustible descargado
Dim usuario As String                   'nombre del usuario
Dim estacion As String                  'nombre de la estacion de servicio
Dim ntelefono As String                 'numero del movil que envia la informacion
Dim telefono As String                  '
Dim unidad As String                    'numero del quipo que hace la descarga
Dim hinicio As String                   'hora de inicio de la descarga
Dim hfin As String                      'hora de finalizacion de la descarga
Dim contador1 As Integer                'numero de sms leidos
Dim contador3 As Integer                '
Dim b As Integer                        ' indica cuando se ha activado la recepcion de sms
Dim X As Integer                        'puntero hoja de recepcion
Dim Y As Integer                        'puntero hoja de base
Dim etiqueta1 As Integer
Dim etiqueta2 As Integer
Dim marca As Integer
Dim MessageType As Long

'*********************************************************************

Private Sub Form_Load()
'----------Inicializacion de las variables----------------------------
    Dim lDeviceCount As Long
    Dim i As Long
    b = 1                               'indica que no se ha activado la recepcion
    etiqueta1 = 0
```

```
    etiqueta2 = 0
    etiqueta3 = 0
    contador1 = 0
    contador3 = 0
    X = 0
    Y = 0
    darth = 0

' Llamada de las funciones especiales de Gsm.

    Set objGsmOut = CreateObject("ActiveXperts.GsmOut")
    Set objConstants = CreateObject("ActiveXperts.SmsConstants")
    Set objGsmIn = CreateObject("ActiveXperts.GsmIn")

' Obtención de numero de dispositivos y elaboración de un listado

    lDeviceCount = objGsmIn.GetDeviceCount()
    lDeviceCount = objGsmOut.GetDeviceCount()
    For i = 0 To lDeviceCount - 1
    comboDevice.AddItem (objGsmIn.GetDevice(i)) '
    Next

    comboDevice.AddItem ("COM1")            ' Puertos Serie
    comboDevice.AddItem ("COM2")
    comboDevice.AddItem ("COM3")

    comboDevice.ListIndex = 0

    comboSpeed.AddItem ("Default")         ' Configuracion de velocidad standar
    comboSpeed.AddItem ("1200")              '  de moviles GSM o modems comunes
    comboSpeed.AddItem ("2400")
    comboSpeed.AddItem ("9600")
    comboSpeed.AddItem ("19200")
    comboSpeed.AddItem ("38400")
    comboSpeed.AddItem ("57600")
    comboSpeed.AddItem ("115200")

    comboSpeed.ListIndex = 0

    comboStore.AddItem ("SM - SIM Memory")              ' Configuracion del tipo de
    comboStore.AddItem ("ME - Device Memory")           ' lectura que se le realizara al
    comboStore.AddItem ("MT - SIM & Device Memory")                ' movil.

    comboStore.ListIndex = 0

End Sub

'*********************************************************************


Private Sub buttonStart_Click()

If b = 1 Then
    fecha = Date                                ' guarda la fecha
    b = 2                                       'indica que se ha activado la recepcion
    Timer1.Enabled = True                       ' activa Timer de monitoreo
    buttonStart.Enabled = False
```

```
   Command1.Enabled = True
   Command2.Enabled = True
   Command5.Enabled = True
   etiqueta3 = comboStore.ListIndex

'--- Verifica que tipo de lectura se esta hacienda y abre las hojas de Excel Base y Recepcion sin que sean visibles.

            If etiqueta3 > 0 Then
             Set appxls1 = CreateObject("Excel.Application")
             Set data1 = appxls1.Workbooks.Open(App.Path & "\Recepcion.xls")
              appxls1.Visible = False
             Set appxls2 = CreateObject("Excel.Application")
             Set data2 = appxls2.Workbooks.Open(App.Path & "\Base.xls")
             appxls2.Visible = False
             End If
  End If
End Sub

'*********************************************************************

Private Sub Timer1_Timer()
Dim i As Integer
hora = Time()                                   'Guarda el tiempo
fecha = Date & "[ED]"              'coloca la fecha y [ED] que indica finalización del día

'---Verifica si se ha llegado al final del dia para guardar la tabla actual y abrir una nueva.

If hora = "11:59:59 p.m." Then
            If etiqueta3 > 0 Then
            appxls1.ActiveWorkbook.SaveAs (App.Path & "\" & fecha)
            appxls1.Quit
            End If
            For i = 0 To 100
            i = i + 1
            Next
                        If etiqueta3 > 0 Then
                        Set appxls1 = CreateObject("Excel.Application")
                        Set data1 = appxls1.Workbooks.Open(App.Path & "\Recepcion.xls")
                        appxls1.Visible = True
                        X = 0
                        Y = 0
                        End If
End If

'El contador 3 nos indica cada cuanto tiempo refrescaremos la tabla para este caso se hace cada 25 segundos ya que el
times esta colocado cada segundo.

If contador3 = 25 Then
smsreciver                                      'llamada de la funcion smsreceiver
contador3 = 0                                   'Inicializando el contador para otra actualizacion.
End If
contador3 = contador3 + 1               ' incremento del contador
End Sub

'*********************************************************************

Private Sub smsreciver()
```

```
Dim NumMessages As Long
Dim i As Long
   Screen.MousePointer = vbHourglass                    'Configuracion del mouse
   ListView.ListItems.Clear                             'limpia los items
   objGsmIn.Device = comboDevice.Text                ' establece el dispositivo entrada
   objGsmOut.Device = comboDevice.Text               ' establece el dispositivo salida
   If comboSpeed.Text = "Default" Then               ' establece la velocidad conexion
      objGsmIn.DeviceSpeed = 0
   Else
      objGsmIn.DeviceSpeed = comboSpeed.Text
   End If


'establece que tipo de lectura se realizara al movil.
objGsmIn.Storage = comboStore.ListIndex
objGsmIn.DeleteAfterReceive = checkDelete.Value     ' Borra los mensajes recibidos.
   objGsmIn.Receive                                         ' Mensajes recibidos

If GetResult = 0 Then                                       ' verificacion de errores
      objGsmIn.GetFirstMessage

          While GetResult = 0                                   'si no hay errores procede a leerlos
         Dim lList As ListItem

' Coloca los mensajes leidos en una lista
         Set lList = ListView.ListItems.Add(, , objGsmIn.MessageTime)
          lList.SubItems(1) = objGsmIn.MessageSender       ' numero telefónico de Tx
          lList.SubItems(2) = objGsmIn.MessageData         ' mensaje de texto
          objGsmIn.GetNextMessage
          appxls1.Worksheets("hoja1").Activate          'Activación de la hoja de calculo
          telefono = objGsmIn.MessageSender        'Extrae el numero telefónico de Tx
          mensaje = objGsmIn.MessageData                    ' Extrae el mensaje
          imei = Mid(mensaje, 5, 14)                        'Extrae del mensaje el imei
          usuario = Mid(mensaje, 28, 4)               'Extrae del mensaje el usuario
          tipo = Mid(mensaje, 1, 4)                         'Extrae del mensaje el tipo
          estacion = Mid(mensaje, 19, 5) 'Extrae del mensaje la estacion de servicio
          descarga = Mid(mensaje, 24, 4)              'Extrae del mensaje la descarga realizada
          hinicio = Mid(mensaje, 32, 28)              'Extrae del mensaje la hora de inicio
          hfin = Mid(mensaje, 60, 28)                 'Extrae del mensaje la hora de finalizacion
          contador1 = contador1 + 1                         ' Incrementa el contador
          data_base                                   'Llamada de la funcion data_base
          Wend
   End If

   Screen.MousePointer = vbArrow
   buttonStart.Enabled = True
End Sub

'**********************************************************************
' Esta función es la encargada de llenar la hoja electrónica recepción con los datos recibidos.

Private Sub data_base()

'—Coloca el contador en la recepción
appxls1.Worksheets("hoja" & Trim(1)).Range("a" & Trim(3 + X)).Value = contador1
```

' Verifica el numero de telefono de un Tx y lo compara con los que estan en la hoja electronica base si existe verifica luego el imei del mismo movil con el de la base si existe procesa la información y la coloca en la tabla de recepcion.

```
Do While appxls2.Worksheets("hoja" & Trim(1)).Range("b" & Trim(3 + Y)).Value <> ""
If appxls2.Worksheets("hoja" & Trim(1)).Range("b" & Trim(3 + Y)).Value = telefono Then
     If appxls2.Worksheets("hoja" & Trim(1)).Range("a" & Trim(3 + Y)).Value = imei Then
       appxls1.Worksheets("hoja" & Trim(1)).Range("b" & Trim(3 + X)).Value = imei
       appxls1.Worksheets("hoja" & Trim(1)).Range("d" & Trim(3 + X)).Value =
       appxls2.Worksheets("hoja" & Trim(1)).Range("c" & Trim(3 + Y)).Value
       unidad = appxls2.Worksheets("hoja" & Trim(1)).Range("d" & Trim(3 + Y)).Value
       appxls1.Worksheets("hoja" & Trim(1)).Range("e" & Trim(3 + X)).Value = unidad
       appxls1.Worksheets("hoja" & Trim(1)).Range("k" & Trim(3 + X)).Value =
      "No existe ninguna anomalia"
       appxls1.Worksheets("hoja" & Trim(1)).Range("h" & Trim(3 + X)).Value = descarga
       appxls1.Worksheets("hoja" & Trim(1)).Range("c" & Trim(3 + X)).Value = telefono
       appxls1.Worksheets("hoja" & Trim(1)).Range("i" & Trim(3 + X)).Value = hinicio
       appxls1.Worksheets("hoja" & Trim(1)).Range("j" & Trim(3 + X)).Value = hfin
       ntelefono = telefono
       appxls1.Worksheets("hoja" & Trim(1)).Range("e" & Trim(3 + X)).Value = unidad

'Decodifica el tipo de combustible que ha sido descargado
' Si RSD1= Diesel,  RSD2= Regular,     RSD3=Super
              If tipo = "RSD1" Then
                appxls1.Worksheets("hoja" & Trim(1)).Range("g" & Trim(3 + X)).Value =
                   "Diesel"
              End If
              If tipo = "RSD2" Then
                appxls1.Worksheets("hoja" & Trim(1)).Range("g" & Trim(3 + X)).Value =
                  "Regular"
               End If
                 If tipo = "RSD3" Then
                appxls1.Worksheets("hoja" & Trim(1)).Range("g" & Trim(3 + X)).Value =
                  "Super"
               End If
       etiqueta1 = 1                        'Indica que no existe problema con el movil
   End If
 End If
 Y = Y + 1
 Loop
Y = 0

 If etiqueta1 = 0 Then                      'Indica que el movil no pertenece a la red
       appxls1.Worksheets("hoja" & Trim(1)).Range("k" & Trim(3 + X)).Value =
          "El  telefono no pertenece a la red"
       etiqueta1 = 0                                'Lo coloca nuevamente sin error
       End If

 ' colocamos el tipo de mensaje a enviar
   MessageType = objConstants.asMESSAGETYPE_UNICODE
   objGsmOut.MessageRecipient = "+503" & ntelefono       ' coloca el numero

' coloamos el mensaje de respuesta para el movil que hizo la descarga exitosa
   objGsmOut.MessageData = "Succes" & estacion
   objGsmOut.MessageType = MessageType                  ' configuramos el mensaje
   objGsmOut.Send                                                  ' Envio de mensaje

     Y = 0                                  ' Se reinicia el puntero
```

```
    X = X + 1                                   ' incremento de puntero

' Se inicializan las variables para evitar errores
telefono = 0
imei = 0
usuario = 0
tipo = 0
estacion = 0
descarga = 0

End Sub

'*********************************************************************
' Funcion que detecta los errores de conexión, de envio y recepcion.

Private Function GetResult() As Long
   Dim lResult As Long
   lResult = objGsmIn.LastError                 ' Obtener el ultimo error
   If (lResult = 0) Then
      textResult.Caption = "Transferencia exitosa"
   Else
If lResult <> 23140 Then
Timer1.Enabled = False
viewbutton.Enabled = False
openbutton.Enabled = False
stopbutton.Enabled = False
buttonStart.Enabled = True
   appxls1.Quit
   appxls2.Quit
   MsgBox ("Error!!! Las configuraciones del dispositivo estan erroneas o el movil no esta conectado")
      textResult.Caption = "ERROR " & lResult & " : " & objGsmIn.GetErrorDescription(lResult)          ' coloca
resiltado del error
   End If
   End If
GetResult = lResult
End Function

'*********************************************************************

Private Sub viewbutton_Click()
If b = 2 Then                                   ' Verifica si la recepcion esta activa
appxls1.Visible = True                      'Hace visible la tabla actual
End If

If b = 1 Then                                   ' Si no esta activa coloca mensaje de error
MsgBox ("Debe activar la recepcion primero")
End If
End Sub

'*********************************************************************

Private Sub savebutton_Click()
If b = 2 Then
fecha = Date & "[" & darth & "]"          'Le coloca a la fecha el numero de modificacion
appxls1.ActiveWorkbook.SaveAs (App.Path & "\" & fecha) 'Guarda la hoja de
                    calculo con la fecha y la condicion de cuantas veces a sido modificada
darth = darth + 1                           ' Incrementa la condicion
```

```
MsgBox ("La tabla se a guardado como" & fecha)
End If
If b = 1 Then                                                    'Verifica si esta activa la recepcion
MsgBox ("Debe activar la recepcion primero")
End If
End Sub

'**********************************************************************
Private Sub openbutton _Click()

' Con la ayuda de un common dialog se habre una ventana para ver los archivos existentes .
Dim sFile As String
   With dlgCommonDialog
      .DialogTitle = "Abrir"
      .CancelError = False
       .Filter = "Todos los archivos (*.xls)|*.xls"
      .ShowOpen
      If Len(.FileName) = 0 Then
         Exit Sub
      End If
      sFile = .FileName
   End With

 Set appxls3 = CreateObject("Excel.Application")
   Set data3 = appxls3.Workbooks.Open(sFile)
   appxls3.Visible = True
End Sub

'**********************************************************************

Private Sub sender_Click()
Transmisor.Show
End Sub

'**********************************************************************

Private Sub stopbutton_Click()
If b = 2 Then                                    ' verifica si esta activa la recepcion
b = 1
Timer1.Enabled = False                  ' detiene el timer receptor
Command1.Enabled = False
Command2.Enabled = False
Command5.Enabled = False
buttonStart.Enabled = True
If etiqueta3 > 0 Then
fecha = Date & "[" & darth & "]"
appxls1.ActiveWorkbook.SaveAs (App.Path & "\" & fecha)
'Guarda la tabla que se estaba llenando.
darth = darth + 1
appxls1.Quit
appxls2.Quit
End If
End If
End Sub

'**********************************************************************
```

```
Private Sub abrir_Click()
Dim sFile As String
   With dlgCommonDialog
      .DialogTitle = "Abrir"
      .CancelError = False
       .Filter = "Todos los archivos (*.xls)|*.xls"
      .ShowOpen
      If Len(.FileName) = 0 Then
         Exit Sub
      End If
      sFile = .FileName
   End With

 Set appxls4 = CreateObject("Excel.Application")     'ejecutarlo
   Set data4 = appxls3.Workbooks.Open(sFile)
   appxls4.Visible = True
End Sub

'*********************************************************************

Private Sub guardar_Click()
Dim sFile As String
  With dlgCommonDialog
      .DialogTitle = "Guardar"
      .CancelError = False
       .Filter = "Todos los archivos (*.xls)|*.xls"
      .ShowSave
      If Len(.FileName) = 0 Then
         Exit Sub
      End If
       sFile = .FileName
   End With
   appxls1.ActiveWorkbook.SaveAs (sFile)
End Sub

'*********************************************************************
Private Sub salir_Click()
If b = 2 Then
Timer3.Enabled = False
appxls2.Quit
fecha = Date & "[" & darth & "]"
appxls1.ActiveWorkbook.SaveAs (App.Path & "\" & fecha)
appxls1.Quit
End If
End
End Sub
```